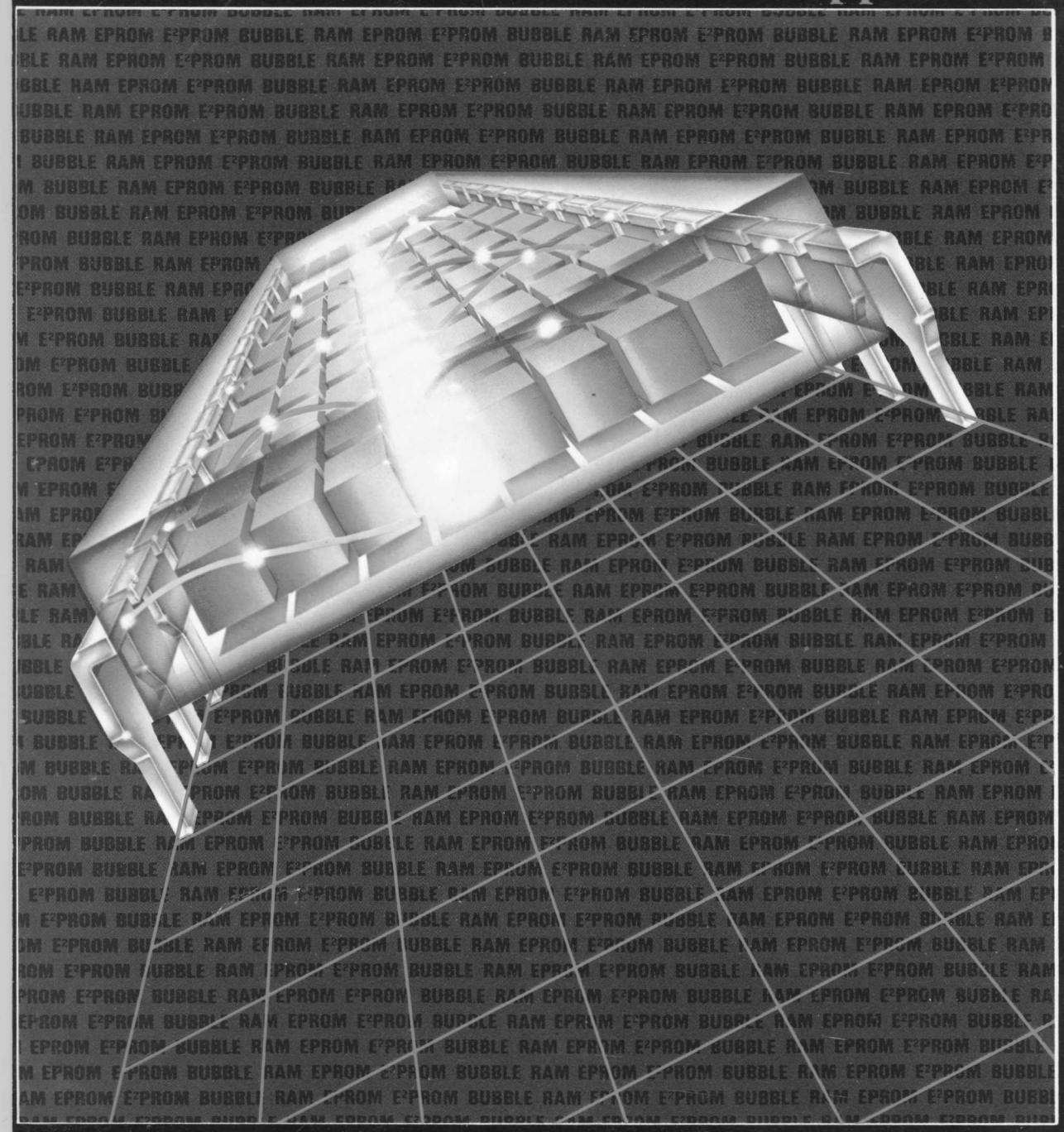
intel

Memory Components Handbook

Supplement



Order Number: 230663-001

LITERATURE

1983 will be a year of transition for Intel's catalog program. In order to better serve you, we are reorganizing many of our catalogs to more closely reflect product groups.

In addition to the new product line handbooks listed below, the INTEL PRODUCT GUIDE (Order No. 210846) is available free of charge. This GUIDE serves as a handy reference tool to Intel's complete product line along with information on quality/reliability, packaging and ordering, customer training classes and product services.

Consult the INTEL LITERATURE GUIDE (no charge, Order No. 210620) for a complete listing of Intel literature. Write or call the Intel Literature Department, 3065 Bowers Avenue, Santa Clara, CA 95051, (800) 538-1876, or (800) 672-1833 (California only).

HANDBOOKS

Memory Components Handbook (Order No. 210830)
Contains all application notes, article reprints, data sheets and other design information on RAMs, DRAMs, EPROMs, E^2PROMs, Bubble Memories.

Microcontroller Handbook (Order No. 210918)
Contains all application notes, article reprints, data sheets, and other user information on the MCS-48, MCS-51 (8-bit) and the new MCS-96 (16-bit) product families.

Military Handbook (Order No. 210461)
Contains complete data sheets on all military products.

Microprocessor and Peripherals Handbook (Order No. 210844)
Contains data sheets on all microprocessors and peripherals. (Individual User Manuals are also available on the 8085, 8086, 8088, 186, 286, etc.)

Development Systems Handbook (Order No. 210940)
Contains data sheets on development systems and supporting software.

OEM Systems Handbook (Order No. 210941)
Contains all application notes, article reprints and data sheets for OEM boards and systems.

intel

MEMORY COMPONENTS HANDBOOK

SUPPLEMENT

1983

* XENIX is a trademark of Microsoft Corporation.
** Ethernet is a trademark of Xerox Corporation.

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

The following are trademarks of Intel Corporation and may only be used to identify Intel products:

iCS	Intellink	iSDM
iLBX	iOSP	iSXM
iMMX	iRMX	Multichannel
Insite	iSBC	MULTIBUS
Intel	iSBX	MULTIMODULE

and the combinations of ICE, iCS, iSBC, MCS or RMX and a numerical suffix.

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

Table of Contents

RAMs (Random Access Memories)

1

intel

December 1982

The Designer's Guide to iRAMs

Order Number: 210860-001

THE DESIGNER'S GUIDE TO iRAMs

The iRAM is the first of a new generation of VLSI memories; a complete dynamic RAM system on a chip. It combines the advantages of the simple static RAM interface with the high density and low power dissipation of the more economical dynamic RAM. While extraordinarily complex internally (more than 150,000 active elements), the external interface of the iRAM is just slightly different from the interface of the 2K × 8 static RAMs that you have used before. The iRAM can sit in a 28-pin Universal Site. Designs based on the 2186 are fully compatible with EPROMs, EEPROMs, and 8K × 8 static RAMs. This is an important consideration when designing for compatibility with multiple vendors. There are three differences between the iRAM and SRAM interface. These are described below.

This guide is intended to show you how to use the iRAM. It contains a functional summary of the iRAM, and several simple microprocessor and microcontroller application examples. The enclosed access time matrix matches the access speeds of the iRAM to the operating speed of any Intel microprocessor. For more detailed design information, please consult Intel Application Note 132 "Designing Memory Systems with the 8K × 8 iRAM".

FUNCTIONAL DESCRIPTION

Just like a static RAM (or EPROM), access to the iRAM is initiated by activating the Chip Enable control signal ($\overline{CE}$). The bar over $\overline{CE}$ indicates that it is an active low signal. Activating $\overline{CE}$ latches the valid external addresses from the system bus into the RAM. In simple systems, the iRAM's address latches can eliminate the need to demultiplex the address/data bus from the processor.

Because the leading (falling) edge of $\overline{CE}$ starts the internal sequencing of a memory cycle, a transition on the $\overline{CE}$ control signal must be glitch-free. Any spurious transitions of $\overline{CE}$ may inadvertantly select the iRAM at the wrong time, resulting in a loss of data or worse. Figure 1 shows a simple circuit that ensures a clean transition for $\overline{CE}$.

Once $\overline{CE}$ has been activated, the user may select one of three different cycles; a Read cycle, a Write cycle or a False Memory cycle. For a Read cycle, the read control line, called $\overline{OE}$ (Output Enable), is activated. Data will remain valid

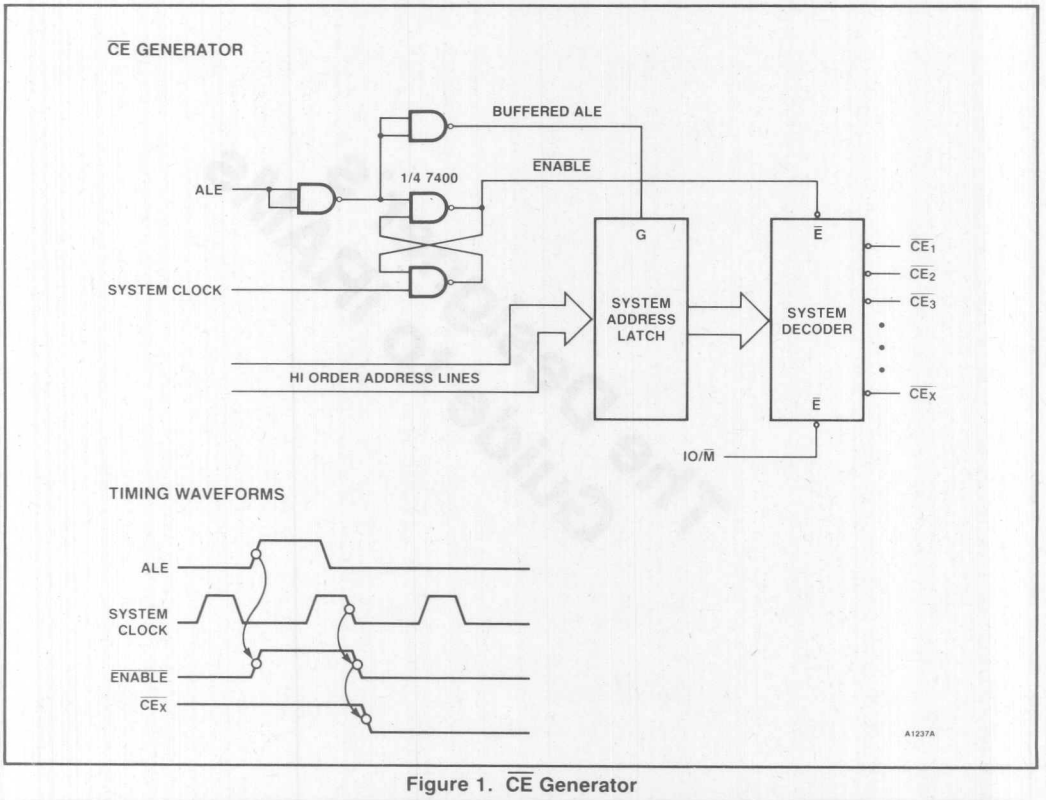


Figure 1. $\overline{CE}$ Generator

at the RAM outputs as long as $\overline{OE}$ is held active — regardless of the state of $\overline{CE}$. $\overline{OE}$ must return to the inactive state prior to the next occurence of $\overline{CE}$.

For a Write cycle, the write control line, called $\overline{WE}$ (Write Enable), is activated after $\overline{CE}$ is enabled. The only requirement for a successful Write cycle is that data be valid at the data inputs prior to $\overline{WE}$ going active. Some older processors, and the iAPX 86/88 in minimum mode, require the addition of a single flip-flop to delay $\overline{WE}$ going active until data is valid. Figure 2 illustrates an example of this circuit and the associated timing waveforms. Finally, $\overline{WE}$ must return to the inactive state prior to the next occurence of $\overline{CE}$. Note that $\overline{OE}$ and $\overline{WE}$ may not both go active during the same cycle.

A False Memory cycle (FMC) occurs whenever $\overline{CE}$ is activated and neither $\overline{OE}$ nor $\overline{WE}$ go active. Addresses must be valid at the iRAM prior to $\overline{CE}$ going active for an FMC. The designer should be aware that some unique timing requirements exist for FMCs.

Because internal refresh may be occurring at any time, a simple handshake procedure is used to notify the processor of a delay in the cycle. If $\overline{CE}$ arrives while the iRAM is in the midst of a refresh cycle, the ready output line (RDY) will go inactive low. RDY is usually used to generate WAIT states, and several RDY lines can be "OR'd" together at the system level. When both the internal refresh cycle and the requested external access cycle are complete, RDY is released and the processor finishes the transfer.

An alternate version of the iRAM gives the designer complete control over access and refresh cycles. On the 2187 iRAM, the RDY output is replaced by the refresh enable input ($\overline{REFEN}$). To initiate a refresh cycle, $\overline{REFEN}$ is activated. An internal refresh row address counter provides the refresh address. Note that $\overline{CE}$ and $\overline{REFEN}$ may not both go active during the same cycle. This synchronous iRAM is especially suited for use with microcontrollers that can not accept a ready handshake line.

In summary, there are three key interface requirements that the iRAM designer must address:

- $\overline{CE}$ input must be glitch free
- $\overline{WE}$ input may have to be delayed until data is valid
- RDY output is used to request a WAIT state during refresh/access overlap.

Some examples of how to use the iRAM are shown on the following pages. A microcontroller application and two different microprocessor systems are outlined, complete with their major interface elements. The timing charts match the operating speed of your processor with the appropriate iRAM access time.

A list of the iRAM literature that is available from Intel is included in this guide. You can get this literature, and answers to any questions about the availability and pricing of the iRAM from your local Intel sales office.

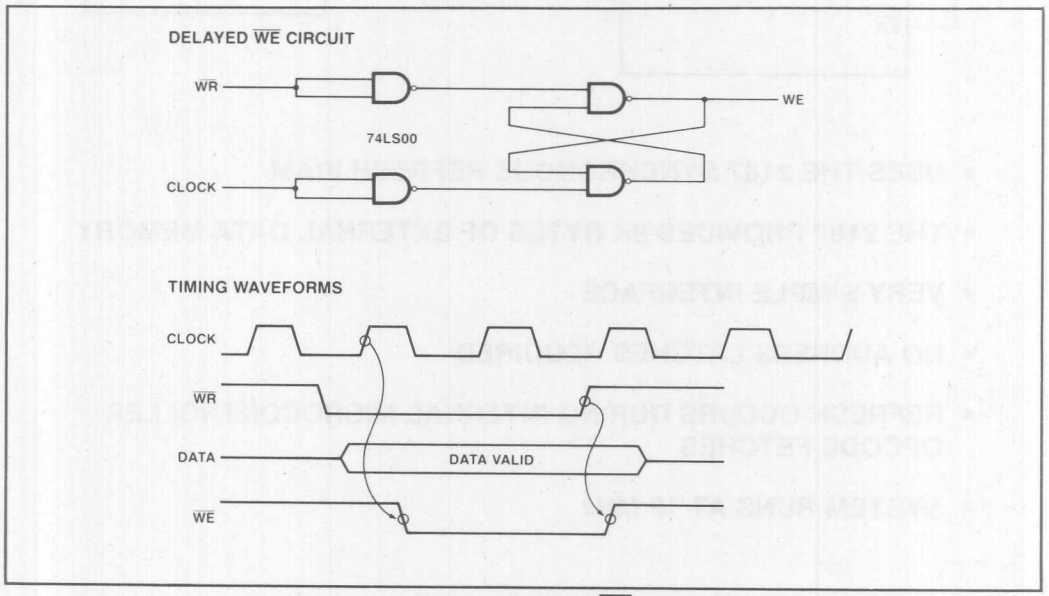
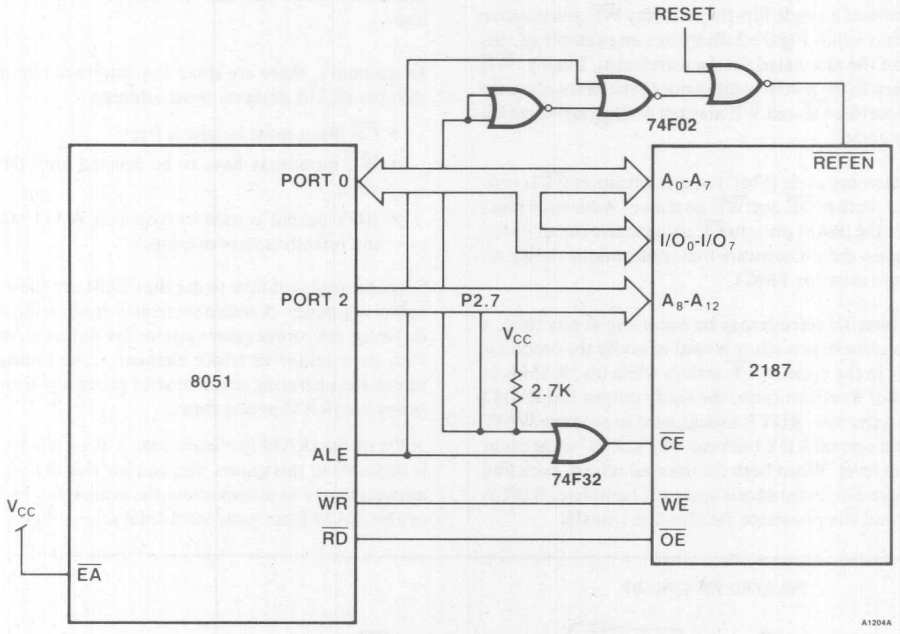


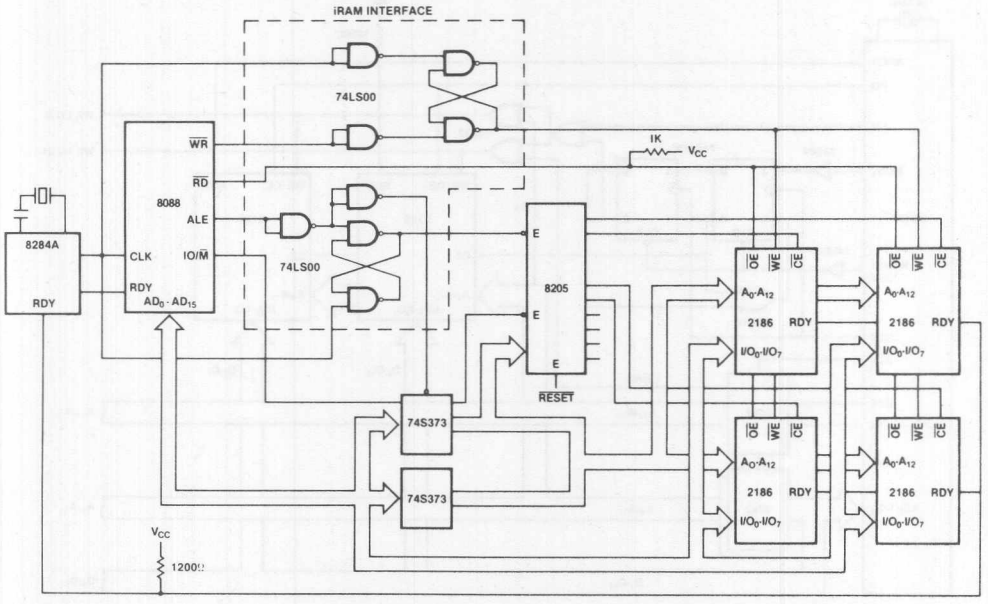
Figure 2. Delayed $\overline{WE}$

8051 MICROCONTROLLER SYSTEM



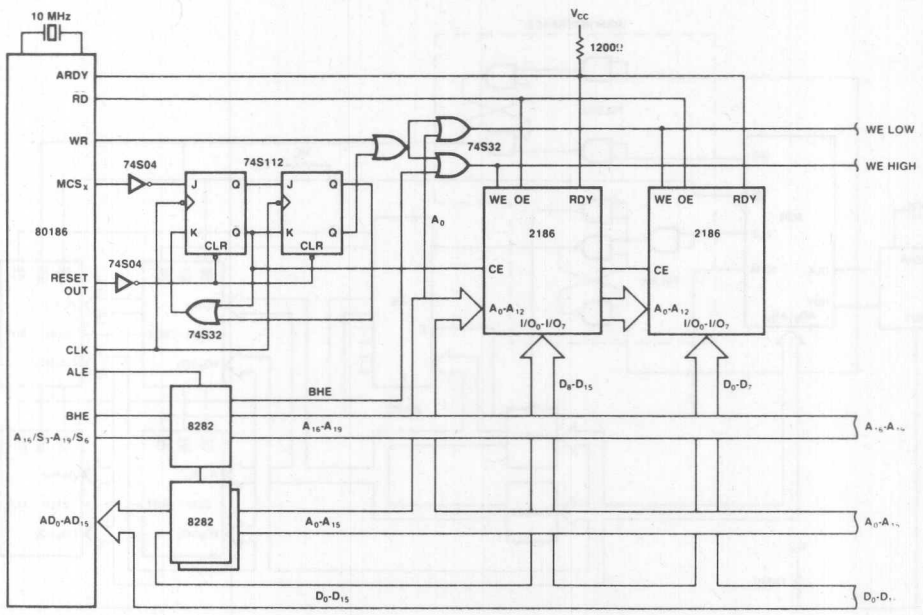
- USES THE 2187 SYNCHRONOUS REFRESH iRAM
- THE 2187 PROVIDES 8K BYTES OF EXTERNAL DATA MEMORY
- VERY SIMPLE INTERFACE
- NO ADDRESS LATCHES REQUIRED
- REFRESH OCCURS DURING INTERNAL MICROCONTROLLER OPCODE FETCHES
- SYSTEM RUNS AT 12 MHz

iAPX 88 MINIMUM MODE SYSTEM



- SIMPLE MINIMUM MODE SYSTEM
- FOUR 2186 iRAMs PROVIDE 32K BYTES OF LOCAL STORAGE
- CLEAN $\overline{CE}$ GENERATED BY ONE TTL PACKAGE (74LS00)
- ONE TTL PACKAGE (74LS00) DELAYS $\overline{WE}$ UNTIL DATA IS VALID
- THE iRAMs ARE IN UNIVERAL SITE SOCKETS — COMPATIBLE WITH SRAMs AND EPROMS
- SYSTEM RUNS AT 5 MHz WITH NO WAIT STATES

iAPX 186 SYSTEM



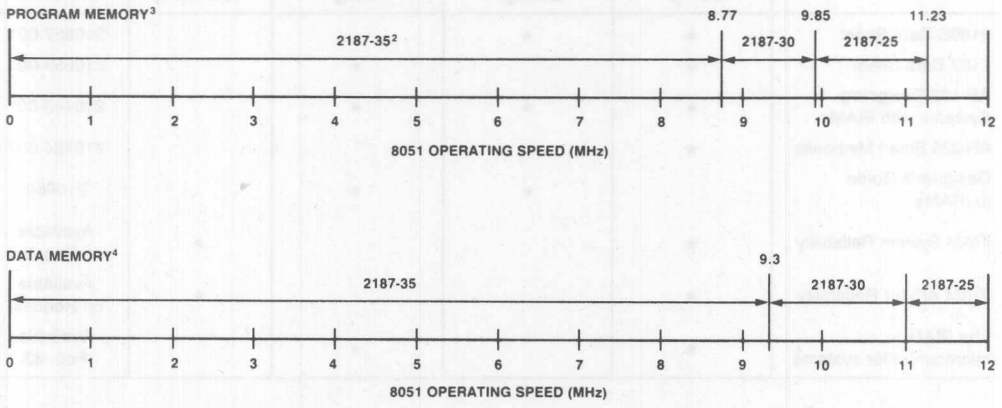
- TWO 2186 iRAMs PROVIDE 8K WORDS OF LOCAL STORAGE
- SIMPLE CIRCUITRY GENERATES A CLEAN $\overline{CE}$
- DELAYED $\overline{WE}$ GATED TO SELECT ONE OR BOTH iRAMs FOR 8 OR 16 BIT DATA TRANSFERS
- READY HANDSHAKE LINE (RDY) CONNECTS DIRECTLY TO THE PROCESSOR
- THE iRAMs UNIVERSAL SITE SOCKETS ARE COMPATIBLE WITH SRAMs AND EPROMs

Microprocessor Operating Speed Vs. iRAM Access Time[1]

Microprocessor Speed		0 Wait States	1 Wait State	2 Wait States	3 Wait States
8085AH	3 MHz	note 2	2186-35	2186-35	2186-35
	5 MHz		2186-35	2186-35	2186-35
	6 MHz		2186-30	2186-35	2186-35
8086[3]	5 Mhz	2186-30	2186-35	2186-35	2186-35
	8 MHz		2186-30	2186-35	2186-35
	10 MHz		2186-25	2186-35	2186-35
80186[4]	8 MHz		2186-30	2186-35	2186-35
	10 MHz			2186-30	2186-35
	15 MHz				2186-25
80286[4]	8 MHz			2186-30	2186-35
	10 MHz			2186-25	2186-35
	15 MHz				

NOTES:
1. Buffer delays not included.
2. Due to its RDY response requirements, the 8085 cannot run without wait states.
3. Timing also applicable for the 8088 microprocessor.
4. Specifications for higher clock speeds not available. Memory requirements for 10 MHz and 15 MHz are extrapolated from 8 MHz specifications.

8051 iRAM REQUIREMENTS[1]



NOTES:
1. Buffers delays not included.
2. Runs with no wait states over this speed range.
3. The iRAM is used to store both program code and data.
4. Program code resides in the 8051's internal ROM or in external ROM.

2186/2187 Literature

Title	Description	For Whom
Data Sheets	2186S, 2187	All iRAM users
Designer's Guide to the iRAM	General introduction to iRAM system design, includes system block diagrams and timing matrix: system speed vs. iRAM access time.	All iRAM users
Designing memory systems with the 8K x 8 iRAM	Application Note 132 — describes the design of iRAM memory systems, includes a description of internal iRAM functions.	All iRAM users
The iRAM in microcontroller systems	How to design microcontroller memory systems using the synchronous 2187 iRAM. Available February 83.	All 2187 users
Smart Memories	Article Reprint 235 — overview of latest trends in smart memory technologies	All iRAM users
Memory Components Handbook	Data sheets and Ap Notes for all Intel Memory Components — includes 2186S, 2187 data sheets and AP 132	All iRAM users
iRAM System Reliability	A summary of the system testing performed on the 2186. Includes reliability data and test descriptions.	iRAM users who require extensive qualification
iRAM Arbiter Reliability	Report on the reliability of the iRAM arbitration circuit.	iRAM users who require extensive qualification

2186/2187 Literature Guide by Topic

	Topic				
Title	**iRAM Tech. Design**	**2186 System Design**	**2187 System Design**	**iRAM Reliability**	**Order Number**
2186S Data Sheet	★	★			210857-001
2187 Data Sheet	★		★		210859-001
AP-132 Designing Systems with iRAMs	★	★	★		210443-001
AR-235 Smart Memories	★				210784-001
Designer's Guide to iRAMs		★	★		210860
iRAM System Reliability	★			★	Available on Request
iRAM Arbiter Reliability	★			★	Available on Request
The iRAM in microcontroller systems	★		★		Available Feb. 83

PRELIMINARY

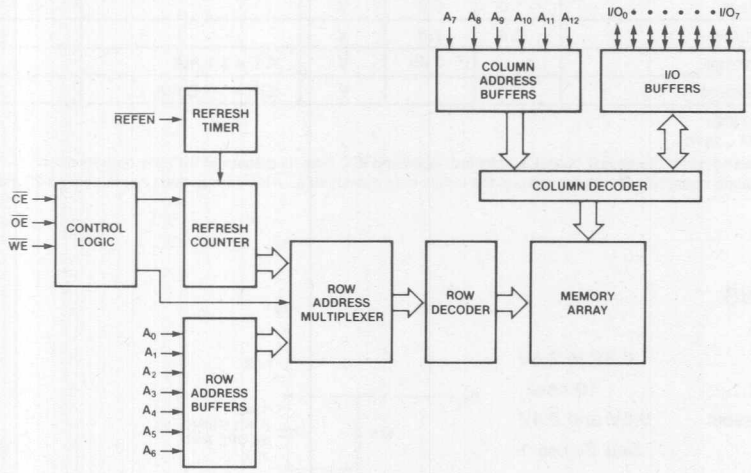
2187A FAMILY
8192 x 8 BIT INTEGRATED RAM

- Low-cost, high-volume HMOS technology
- High density one transistor cell
- Single +5V ± 10% supply
- Proven HMOS Reliability
- Low active current (70 mA)
- Simple synchronous refresh operation
- 2764 EPROM compatible pin-out
- Two-line bus control
- JEDEC standard 28-pin site
- Low standby current (20 mA)

The Intel 2187 is an 8192 word by 8-bit integrated random access memory (iRAM) fabricated on Intel's proven HMOS dynamic RAM technology. Packaged in the industry standard 28-pin DIP, the 2187 conforms to the industry standard JEDEC 28-pin site.

The 2187 is particularly suited for use in microcontroller applications, incorporating many requisite system features. These include low power dissipation, automatic initialization, extended cycle operation and two-line bus control to eliminate bus contention.

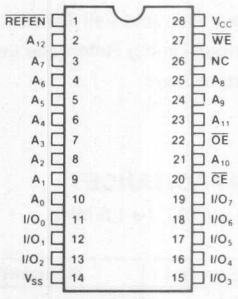
BLOCK DIAGRAM



PIN NAMES

A_0-A_{12}	ADDRESS INPUTS
$\overline{CE}$	CHIP ENABLE
$\overline{OE}$	OUTPUT ENABLE
$\overline{WE}$	WRITE ENABLE
I/O_0-I/O_7	DATA INPUT/OUTPUT
$\overline{REFEN}$	REFRESH ENABLE
V_{CC}	+5V POWER
V_{SS}	GROUND

PIN CONFIGURATION



28-LEAD

April, 1983
Order Number: 210859-001

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias ..... −10°C to +80°C
Storage Temperature ..... −65°C to +150°C
Voltage on Any Pin with
Respect to Ground ..... −1.0V to +7V
D.C. Continuous Current per Output ..... 10 mA
D.C. Maximum Data Out Current ..... 50 mA
D.C. Power Dissipation ..... 1.0 W

** COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

D.C. AND OPERATING CHARACTERISTICS[1]

TA = 0°C to +70°C, VCC = +5V ± 10% unless otherwise noted.

Symbol	Parameter	Limits Min.	Limits Max.	Unit	Test Conditions	Notes
ILI	Input Load Current (All Input Pins)		10	μA	V_{IN} = VSS to VCC	
|ILO|	Output Leakage Current		10	μA	$\overline{OE}$ = VIH	
ICC	Operating Current		70	mA	Minimum Cycle Time	2
ISB	Standby Current		20	mA	$\overline{CE}$ = VIH	
VIL	Input Low Voltage	−1.0	0.8	V		3
VIH	Input High Voltage	2.4	7.0	V		
VOL	Output Low Voltage		0.45	V	IOL = 2.1 mA	
VOH	Output High Voltage	2.4		V	IOH = −1.0 mA	

NOTES FOR D.C. CHARACTERISTICS:
1. Typical limits are VCC = +5V, TA = 25°C.
2. ICC is dependent on outputs loading when the device output is selected. Specified ICC max. is measured with the outputs open.
3. Specified VIL min. is for steady state operation. During transitions the inputs may overshoot to −2.0V for periods not to exceed 20 nsec.

A.C. TEST CONDITIONS

Input Pulse and Timing
Reference Levels ..... 0.8V to 2.4V
Input Rise and Fall Times ..... 10 nsec.
Output Timing Reference Levels .... 0.6V and 2.4V
Output Load ..... See Figure 1

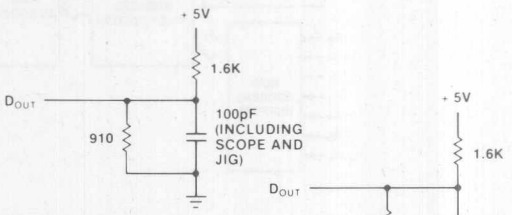


Figure 1.

Figure 2.

(FOR HIGH IMPEDANCE MEASUREMENTS ONLY)

CAPACITANCE[4]

TA = 25°C, f = 1.0 MHz

Symbol	Parameter	Max.	Unit	Conditions
C_{ADD}	Address Capacitance	8	pF	V_{ADD} = 0V
$C_{I/O}$	I/O Capacitance	14	pF	$V_{I/O}$ = 0V
C_{IN}	Control Capacitance	14	pF	V_{IN} = 0V

NOTE: 4. This parameter is characterized and not 100% tested.

A.C. CHARACTERISTICS

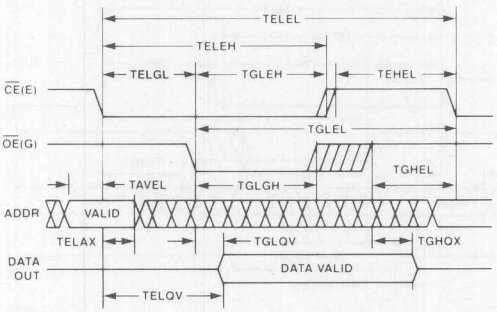
TA = 0°C to +70°C, VCC = +5V ± 10% unless otherwise noted

READ CYCLE ($\overline{WE}$ = VIH)

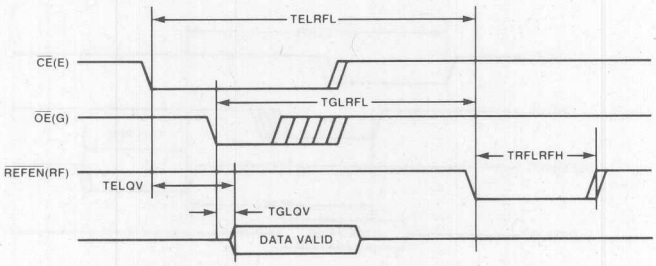
Symbol	Parameter	2187-25		2187-30		2187-35		Unit	Notes
		Min.	Max.	Min.	Max.	Min.	Max.		
TELEL	Cycle Time	425		500		600		ns	
TELQV	Access Time from $\overline{CE}$		250		300		350	ns	
TGLQV	Access Time from $\overline{OE}$		65		70		75	ns	
TELEH	$\overline{CE}$ Pulse Width	40		40		40		ns	
TEHEL	$\overline{CE}$ High Time	40		40		40		ns	
TAVEL	Address Set-Up Time	0		0		0		ns	
TELAX	Address Hold Time	35		35		35		ns	
TGLEL	$\overline{OE}$ low to next $\overline{CE}$ low	250		275		300		ns	
TGLGH	$\overline{OE}$ Pulse Width	65		70		75		ns	
TGHEL	$\overline{OE}$ high to next $\overline{CE}$ low	40		40		40		ns	
TGHQX	$\overline{OE}$ high to Data Float	0	70	0	70	0	70	ns	1
TELGL	$\overline{CE}$ low to $\overline{OE}$ low		10,000		10,000		10,000	ns	
TGLEH	$\overline{OE}$ low to $\overline{CE}$ high	40		40		40		ns	
TELRFL	$\overline{CE}$ low to $\overline{REFEN}$ low	425		500		600		ns	
TGLRFL	$\overline{OE}$ low to $\overline{REFEN}$ low	250		275		300		ns	
TRFLRFH	$\overline{REFEN}$ Pulse Width	70	9800	70	9800	70	9800	ns	5

WAVEFORMS

READ CYCLE



READ CYCLE FOLLOWED BY REFRESH



A.C. CHARACTERISTICS

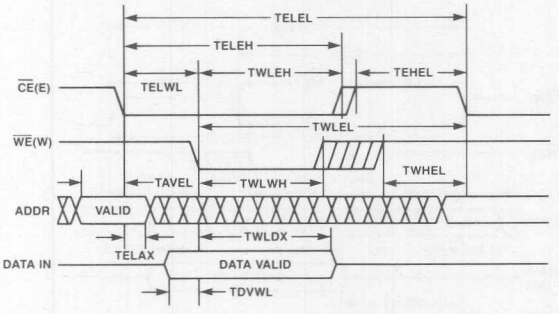
TA = 0°C to +70°C, VCC = +5V ± 10% unless otherwise noted.

WRITE CYCLE ($\overline{OE}$ = VIH)

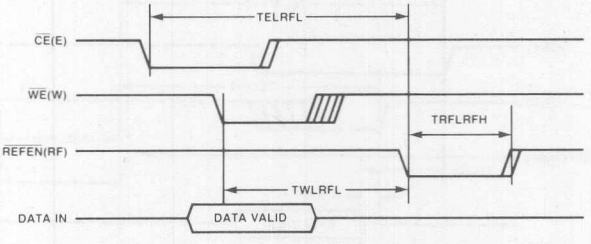
Symbol	Parameter	2187-25		2187-30		2187-35		Unit	Notes
		Min.	Max.	Min.	Max.	Min.	Max.		
TELEL	Cycle Time	425		500		600		ns	
TELEH	$\overline{CE}$ Pulse Width	40		40		40		ns	
TEHEL	$\overline{CE}$ High Time	40		40		40		ns	
TAVEL	Address Set-Up Time	0		0		0		ns	
TELAX	Address Hold Time	35		35		35		ns	
TWLEL	$\overline{WE}$ low to next $\overline{CE}$ low	250		300		350		ns	
TWLWH	$\overline{WE}$ Pulse Width	40		40		40		ns	
TWHEL	$\overline{WE}$ high to next $\overline{CE}$ low	40		40		40		ns	
TDVWL	Data Set-Up to $\overline{WE}$ low	0		0		0		ns	
TWLDX	Data Hold from $\overline{WE}$ low	40		45		50		ns	
TELWL	$\overline{CE}$ low to $\overline{WE}$ low		10,000		10,000		10,000	ns	
TWLEH	$\overline{WE}$ low to $\overline{CE}$ high	40		40		40		ns	
TELRFL	$\overline{CE}$ low to $\overline{REFEN}$ low	425		500		600		ns	
TWLRFL	$\overline{WE}$ low to $\overline{REFEN}$ low	250		300		350		ns	
TRFLRFH	$\overline{REFEN}$ Pulse Width	70	9800	70	9800	70	9800	ns	5

WAVEFORMS

WRITE CYCLE



WRITE CYCLE FOLLOWED BY REFRESH



A.C. CHARACTERISTICS

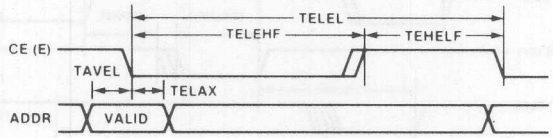
TA = 0°C to +70°C, VCC = +5V ± 10% unless otherwise noted.

FALSE MEMORY CYCLE ($\overline{OE}$ and $\overline{WE}$ = VIH)

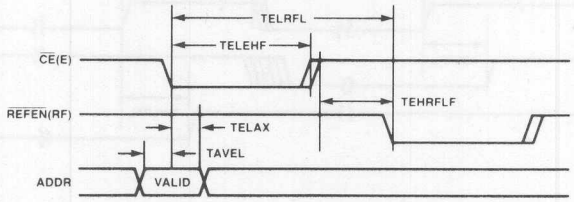
Symbol	Parameter	2187-25		2187-30		2187-35		Unit	Notes
		Min.	Max.	Min.	Max.	Min.	Max.		
TELEL	Cycle Time	425		500		600		ns	
TELEHF	$\overline{CE}$ Pulse Width	40	10,000	40	10,000	40	10,000	ns	2
TEHELF	$\overline{CE}$ High Time during F.M.C.	200		250		275		ns	3
TAVEL	Address Set-Up Time	0		0		0		ns	
TELAX	Address Hold Time	35		35		35		ns	
TELRFL	$\overline{CE}$ low to $\overline{REFEN}$ low	425		500		600		ns	
TEHRFLF	$\overline{CE}$ high to $\overline{REFEN}$ low after F.M.C.	200		250		275		ns	

WAVEFORMS

FALSE MEMORY CYCLE



FALSE MEMORY CYCLE FOLLOWED BY A REFRESH



A.C. CHARACTERISTICS

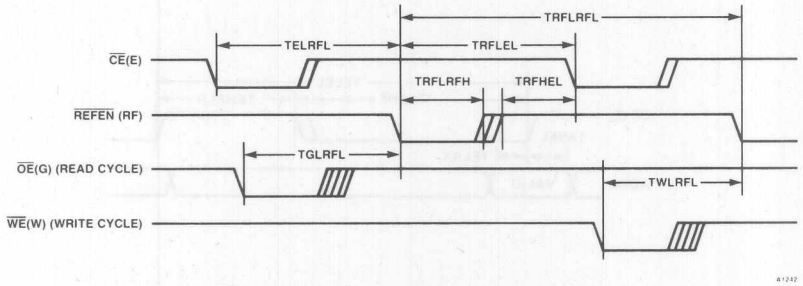
TA = 0°C to +70°C, VCC = +5V ± 10% unless otherwise noted.

REFRESH CYCLE

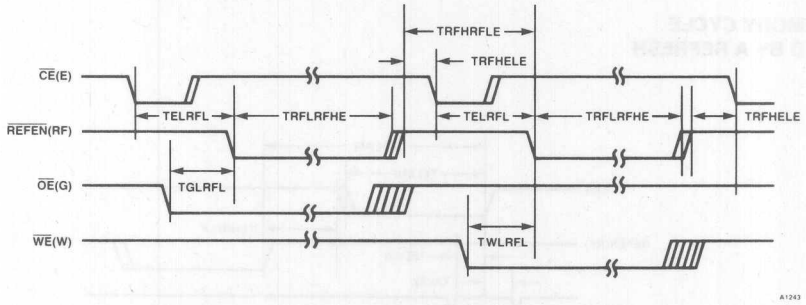
Symbol	Parameter	2187-25		2187-30		2187-35		Unit	Notes
		Min.	Max.	Min.	Max.	Min.	Max.		
TELRFL	$\overline{CE}$ low to $\overline{REFEN}$ low	425		500		600		ns	
TRFLRFH	$\overline{REFEN}$ Pulse Width	70	9800	70	9800	70	9800	ns	5
TRFHEL	$\overline{REFEN}$ high to $\overline{CE}$ low	40		40		40		ns	
TRFLRFL	$\overline{REFEN}$ cycle time to guarantee refresh	350	15600	400	15600	450	15600	ns	
TRFLEL	$\overline{REFEN}$ low to $\overline{CE}$ low	350		400		450		ns	
TRFHELE	$\overline{REFEN}$ high to $\overline{CE}$ low — extended cycle	350		400		450		ns	4
TGLRFL	$\overline{OE}$ low to $\overline{REFEN}$ low	250		275		300		ns	
TWLRFL	$\overline{WE}$ low to $\overline{REFEN}$ low	250		300		350		ns	
TRFLRFHE	$\overline{REFEN}$ Pulse Width — extended cycle	10,000		10,000		10,000		ns	4,5
TRFHRFLE	$\overline{REFEN}$ high to $\overline{REFEN}$ low — extended cycle	425		500		600		ns	

WAVEFORMS

REFRESH CYCLE



EXTENDED CYCLE REFRESH[5]



NOTES FOR A.C. CHARACTERISTICS:

1. Transition is measured ±500 mV from steady state logic level with specified loading in Figure 2.
2. Maximum applies for F.M.C. Only.
3. Note TEHELF > TEHEL.
4. TRFHELE > TRFHEL and TRFLRFHE > TRFLRFH.
5. Extended cycles occur when the $\overline{REFEN}$ pulse width is ≥10 μsec.

FUNCTIONAL DESCRIPTION

The 2187 has four control pins: $\overline{CE}$ (Chip Enable), $\overline{OE}$ (Output Enable), $\overline{WE}$ (Write Enable), and $\overline{REFEN}$ (Refresh Enable). These control lines select and control the operation of the iRAM.

$\overline{CE}$ is the general purpose chip enable line. It is an edge triggered signal that controls several internal operations. An access cycle begins with the leading (falling) edge of $\overline{CE}$. At this time, the external address is latched into the 2187 and is held throughout the current cycle. $\overline{CE}$ may be pulsed or it may remain low throughout the cycle.

$\overline{OE}$ selects a read cycle and controls the output data bus. When $\overline{OE}$ goes active (low) during a read cycle, the output drivers of the 2187 are enabled. Data remains valid on the output lines as long as $\overline{OE}$ is active — independent of the state of $\overline{CE}$.

$\overline{WE}$ is the edge triggered input that defines a write cycle. During write cycles, data is latched from the external bus into the 2187 by the leading (falling) edge of $\overline{WE}$. $\overline{WE}$ and $\overline{OE}$ may not be active during the same cycle.

$\overline{REFEN}$ initiates the internal refresh cycles of the 2187. A refresh cycle begins whenever $\overline{REFEN}$ goes active (low). An internal refresh address counter provides the refresh address. To prevent conflicts between refresh and access cycles, several timing parameters, referenced to $\overline{REFEN}$, must be met. These parameters are specified in the timing tables of the 2187 A.C. Characteristics.

Access Cycles

READ CYCLE

A read cycle is initiated when both $\overline{CE}$ and $\overline{OE}$ go active low during the same cycle. Access times are specified from both $\overline{OE}$ and $\overline{CE}$. After $\overline{OE}$ goes active (low), and the hold time for $\overline{CE}$ has been met (TGLEH), $\overline{CE}$ may go inactive. As long as $\overline{OE}$ remains active (low), data remains on the output lines — independent of the level of $\overline{CE}$.

$\overline{WE}$ may not go active (low) during the read cycle.

WRITE CYCLE

A write cycle occurs when both $\overline{CE}$ and $\overline{WE}$ go active (low) in the same cycle. Once $\overline{WE}$ has gone active (low), $\overline{CE}$ may go inactive after a minimum hold time (TWLEH). The leading (falling) edge of $\overline{WE}$ latches data from the external bus into the 2187. Data must be valid at this time.

$\overline{OE}$ must not go active during the write cycle.

FALSE MEMORY CYCLE

A false memory cycle (FMC) occurs when $\overline{CE}$ goes active (low), and $\overline{OE}$ and $\overline{WE}$ remain inactive (high). No memory cycle is performed, but address set-up and hold times must be met to guarantee the integrity of internal data. During an FMC, the 2187 performs a refresh cycle on the externally addressed row.

Note that some of the $\overline{CE}$ timing specifications for an FMC differ from those of a read or write cycle.

Other Operating Modes

REFRESH OPERATION

The 2187 supports three refresh modes. Two are controlled externally. The third mode can be enabled when the 2187 is not accessed for extended periods of time (during system stand-by or extended cycle operation). An internal refresh timer guarantees refresh to maintain data integrity.

A refresh cycle is initiated by the leading (falling) edge of $\overline{REFEN}$. Once a refresh cycle begins, cycle operation is internal and automatic. $\overline{REFEN}$ may go inactive (high) after the minimum active low pulse time has been met. Addresses are supplied by the internal refresh address counter. To guarantee internal data integrity, $\overline{REFEN}$ must be strobed at least 128 times in every 2 millisecond period. $\overline{REFEN}$ pulses may be distributed or grouped (burst mode).

$\overline{CE}$ may not go active (low) while a refresh cycle is in progress.

The 2187 may also be refreshed by read, write, or false memory cycles. To accomplish this, the user's system must cycle through each of the 128 rows in every 2 millisecond time frame. In this mode, the refresh address is comprised of the seven lowest order address bits (A_0-A_6), supplied externally.

EXTENDED CYCLE OPERATION

Extended cycle operation is useful for single-step operations and is defined by $\overline{OE}$ or $\overline{WE}$ remaining active (low) for indefinite periods of time. ($\overline{CE}$ is allowed to return high). As long as $\overline{OE}$ is active (read cycles only), data will remain valid on the output bus. During write cycles, data is latched on the leading (falling) edge of $\overline{WE}$. Throughout the remainder of the extended cycle, data may change on the external lines without affecting the contents of the iRAM.

To guarantee refresh of the iRAM array during extended cycles, $\overline{REFEN}$ must go active (low) after $\overline{CE}$ has gone active (low). While $\overline{REFEN}$ is low, an internal timer guarantees that the iRAM array is adequately refreshed, thus ensuring data integrity. Refresh cycles will occur automatically, even if $\overline{OE}$ or $\overline{WE}$ remains low. *While $\overline{REFEN}$ is low, $\overline{CE}$ may not go active (low).* Note that if $\overline{REFEN}$ is not enabled (low), proper refresh of the 2187 array cannot be guaranteed during extended cycles.

Once $\overline{REFEN}$ returns inactive (high), there are two timing specifications that must be met before the iRAM is accessed again. These are TRFLEL and TRFHELE.

The internal refresh timer may also be enabled (by holding $\overline{REFEN}$ low) when the iRAM will not be accessed for extended periods of time. It is up to the user to ensure that when $\overline{REFEN}$ returns inactive (high), no timing specifications are violated.

Initialization

Once V_{CC} is within specification, the 2187 is initialized by holding $\overline{REFEN}$ active (low) and all other control inputs inactive (high) for 100 microseconds. This may be done any time after V_{CC} meets specification. Normal operation may begin immediately after initialization.

Interfacing Considerations

The 2187 is ideally suited for use with microcontroller systems that use external memory. Figure 1 is an illustration of the simple circuitry required to interface a 2187 and an 8051 microcontroller. In this particular design, all program memory is located in the 8051's internal ROM. External data memory is mapped into the lower 32K bytes of the 8051 address space.

The 2187 is an edge enabled RAM, therefore a stable $\overline{CE}$ clock is necessary to guarantee proper iRAM operation. The system shown in Figure 1 uses ALE (Address Latch Enable) to generate $\overline{CE}$ for the 2187. Because the 8051 generates ALE for all memory cycles, internal and external, ALE must be gated with an external address line. This ensures that the 2187 is only selected during external cycles, and not during the shorter internal cycles. To guarantee data integrity, the 2187 is refreshed whenever the 8051 performs an internal cycle. In this case, the trailing (falling) edge of ALE is used to generate the $\overline{REFEN}$ signal. Once a refresh cycle begins, it will terminate automatically. This design ensures that the 2187 is properly refreshed and that it will always be ready to respond to access cycles in systems running at speeds up to 12 MHz.

Application note 132 "Designing memory systems with the 8K x 8 iRAM", contains a detailed analysis of the iRAM interface. It also describes several designs that incorporate the iRAM into microprocessor and microcontroller systems.

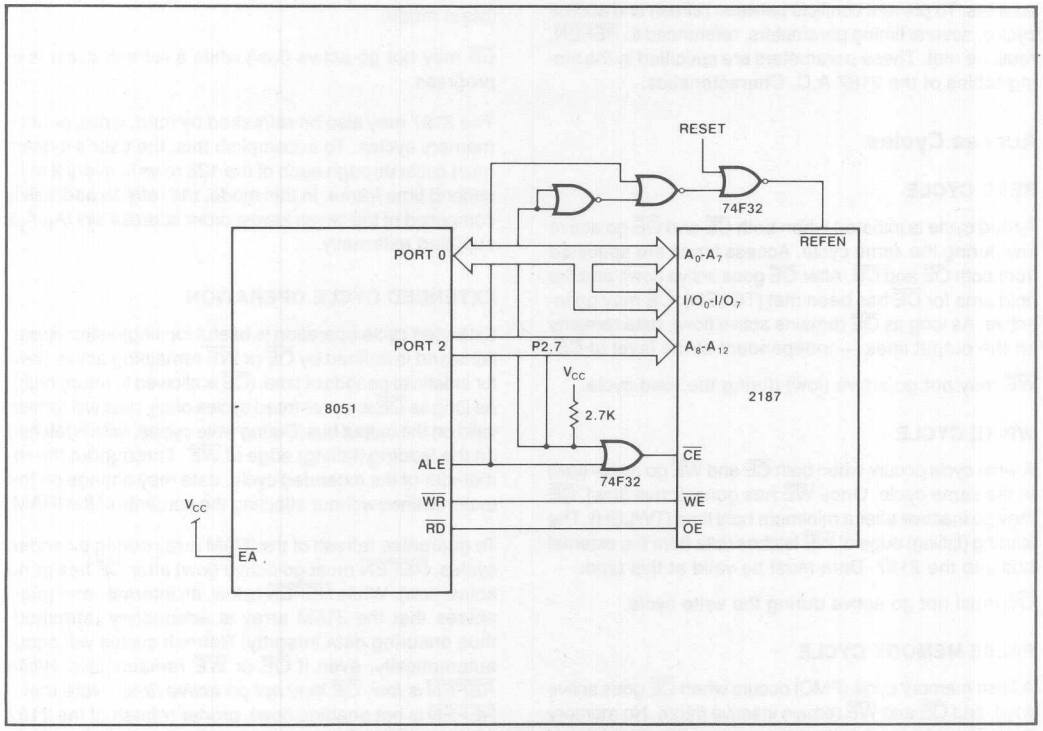


Figure 1. 8051/2187 System Interface

Layout Considerations

To ensure compatibility with other 28-pin memory devices such as EPROMs, several pins require close examination: specifically pins number 1, 26, and 27. Following is a discussion of the system level operation and the design considerations for these pins.

PIN #1

Pin 1 on all EPROMs is reserved for the high voltage programming bias V_{PP}. EPROMS are usually programmed external to the system. Therefore, in normal system operation, pin 1 is connected to V_{CC}.

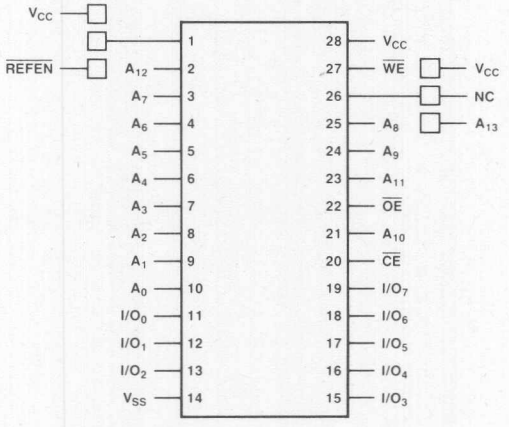
Pin 1 on the 2187 is $\overline{\text{REFEN}}$, the refresh control input. $\overline{\text{REFEN}}$ may come from a microcontroller, or a synchronous refresh timing circuit. In a system incorporating a 28 pin universal site, a trace should be run from pin 1 to the source of the refresh control signal generator, with an alternate trace running to V_{CC}. A jumper option would select $\overline{\text{REFEN}}$ when a 2187 is in the socket. V_{CC} would be selected when the site is occupied by an EPROM. (See Figure 2).

PIN #26

While pin 26 is a No Connect for both the 2186 and the 2764 EPROM, a trace to pin 26 from V_{CC} will guarantee compatibility between 24 pin and 28 pin EPROMs. Pin 26 will carry the additional address bit required to future higher density memories. For flexibility, provide a jumper for an address bit or V_{CC} on pin 26.

PIN #27

Pin 27 is labelled $\overline{\text{WE}}$ on the RAM and $\overline{\text{PGM}}$ on the EPROM. While $\overline{\text{WE}}$ is a system level control signal, $\overline{\text{PGM}}$ is only used when programming the EPROM (V_{PP} at +21V). $\overline{\text{PGM}}$ may be allowed to toggle during normal EPROM operation (V_{PP} at +5V). Therefore, $\overline{\text{WE}}$ may be bussed to every socket location with no jeopardy of illegal operation.



Jumper Matrix Connections

Pin	iRAM	EPROM		
	2187	2732	2764	27128
1	$\overline{\text{REFEN}}$	—	V_{CC}	V_{CC}
26	N.C.	V_{CC}	N.C.	A_{13}

Figure 2. Universal Site Jumper Matrix — iRAM/EPROM Interchange

EPROMs (Erasable Programmable Read Only Memories)

2

intel

APPLICATION NOTE

AP-154

April 1983

Programming Intel's 27256 EPROM

Robert Davis
Applications Engineering
Non-Volatile Memory Division
Intel Corporation

APRIL 1983
ORDER NUMBER: 210970-002

INTRODUCTION

With the introduction of the 27256 32K byte EPROM, a new generation of high density componentized software is possible. Intel's process and product technology advances have increased EPROM memory storage capabilities from 2K bits to 256K bits. This non-volatile memory can allow the designer a more reliable, user-friendly system. Since it is produced in the 28 pin JEDEC-approved Intel universal site, the 27256 can easily be designed into existing printed circuit boards.

Figure 1 shows the evolution of Intel's EPROM family. The new generation 27256 brings with it improved performance and state-of-the-art reliability. The absence of $\overline{PGM}$, replaced by A14, and a lower programming voltage (12.5V) highlight the additions accompanying the 27256. Advanced technology from the 27256 will soon bring enhanced performance to lower density EPROMs, specifically the 2764A and 27128A. This document concerns programming characteristics of the 27256, concentrating on those factors which will be new to the EPROM memory designer.

THE int$_e$ligent Programming™ ALGORITHM

The int$_e$ligent Programming™ Algorithm was developed as an improved alternative to the 50 msec per byte programming techniques for Intel's 2764 and 27128 EPROMs. By taking advantage of the variable programming times required by the cells in an EPROM array, programming speed increases of 5 or 6 times have been achieved. The success of this algorithm, coupled with the long programming times which would be inherent in programming the 27256 with a 50 msec per byte algorithm, has prompted Intel to develop a new 27256 int$_e$ligent Programming™ Algorithm specifically tailored to the requirements of the customer.

The 27256 programming algorithm is similar to the int$_e$ligent Programming Algorithms used for Intel's 2764 and 27128 EPROMs. It is now available as a standard feature in many PROM programmers. This new programming algorithm is fast and guarantees that each cell has been programmed reliably.

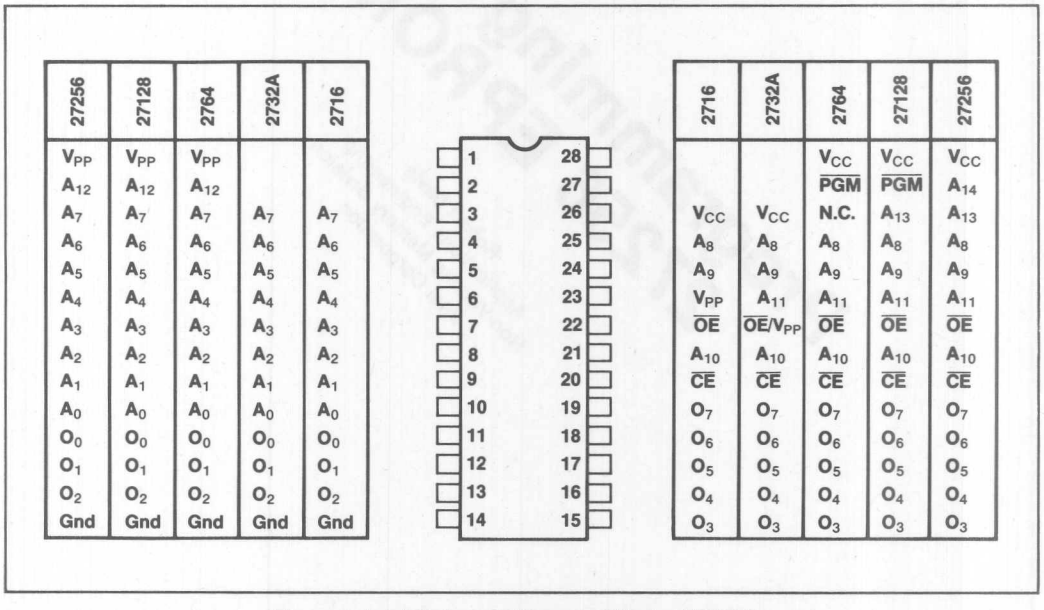


Figure 1. Intel Universal Memory Site for EPROMs

The 27256 int$_e$ligent Programming Algorithm shown in figure 2 is a feedback control loop. In examining this diagram, three distinct characteristics can be seen. First, the programming voltage has been reduced from 21 ± 0.5 volts, as was required on earlier generation 2764 and 27128 EPROMs, to 12.5 ± 0.3 volts. In all cases the V_{PP} voltage should never exceed 13 volts, and a 0.1 microfarad capacitor should be placed between V_{PP} and ground to insure proper decoupling. The technology advances implemented in the 27256 allow the programming voltage to be reduced while taking advantage of the performance of the programmed cell. The second characteristic that should be noted is the maximum number of 1ms pulses which are applied throughout the closed loop programming algorithm. As shown, 25 iterations of the loop is the maximum number allowed. Intel's reliability data indicate that 25 iterations is sufficient to reliably program each of the EPROM bits in the array. The last characteristic which should be mentioned is the insertion of a byte verify after the iteration count has been maximized. This will save time by failing any device which may not verify correctly after 25 one msec pulses.

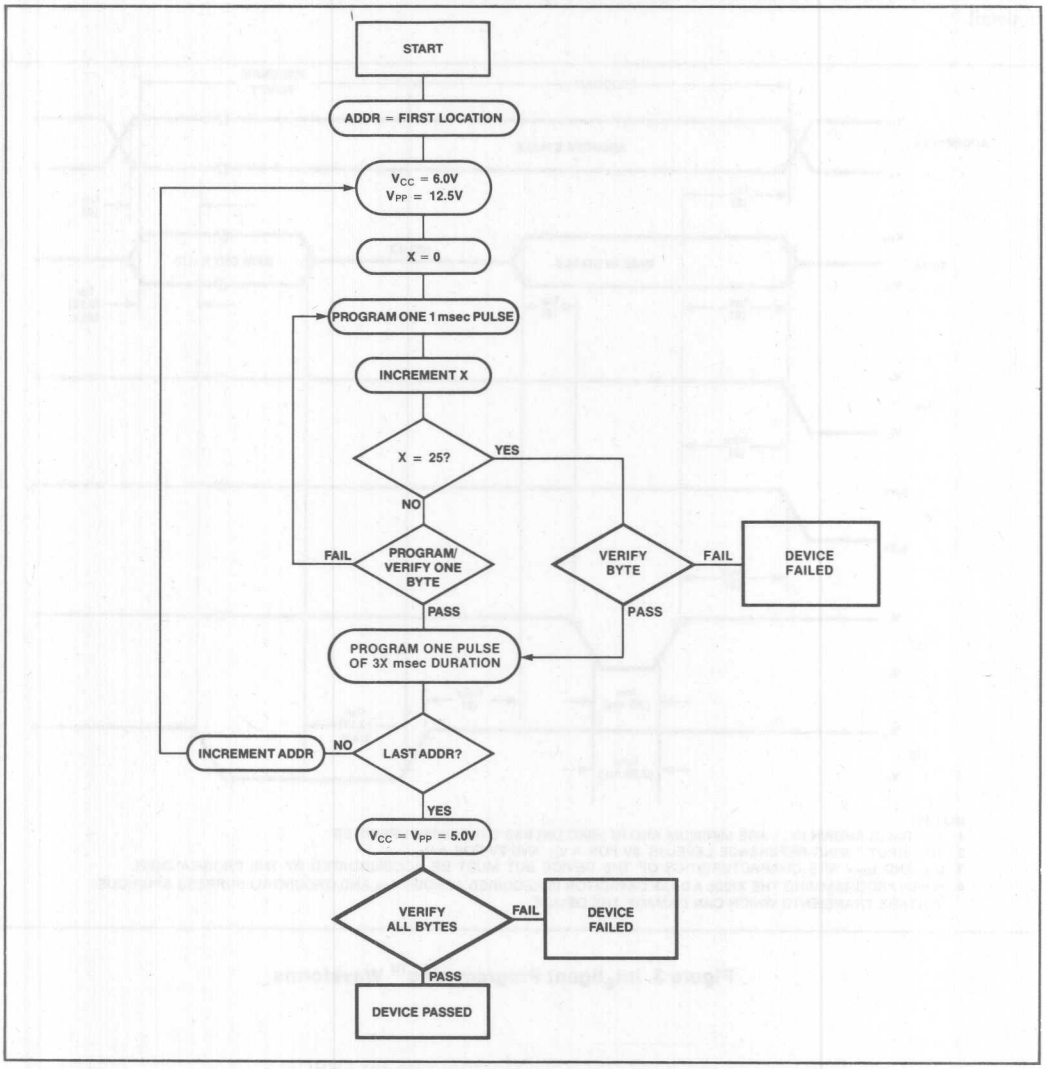


Figure 2. int$_e$ligent Programming™ Algorithm

27256 PROGRAMMING WAVEFORMS

With the replacement of the $\overline{PGM}$ signal by A14 on Pin 27, changes are required in the manner in which the 27256 is programmed. Figure 3 shows the waveforms required to program the 27256. As illustrated, the 27256 will be placed in the program mode when V_{PP} is set to 12.5 volts, and $\overline{CE}$ is pulsed to V_{IL}. The verify operation will then be selected with $\overline{CE}$ at V_{IH}, and $\overline{OE}$ held at V_{IL}. This results from multiplexing both the chip enable function and the programming enable function onto Pin 20. The V_{PP} level, either 5 volts or 12.5 volts, gates which function is selected. Table 1 indicates the various modes of operation of a 27256.

In on-board gang programming applications, care must be taken not to enable the outputs of all resident 27256 devices while in the verify mode. A common $\overline{OE}$ signal could cause bus contention. A separate document will discuss this issue in detail.

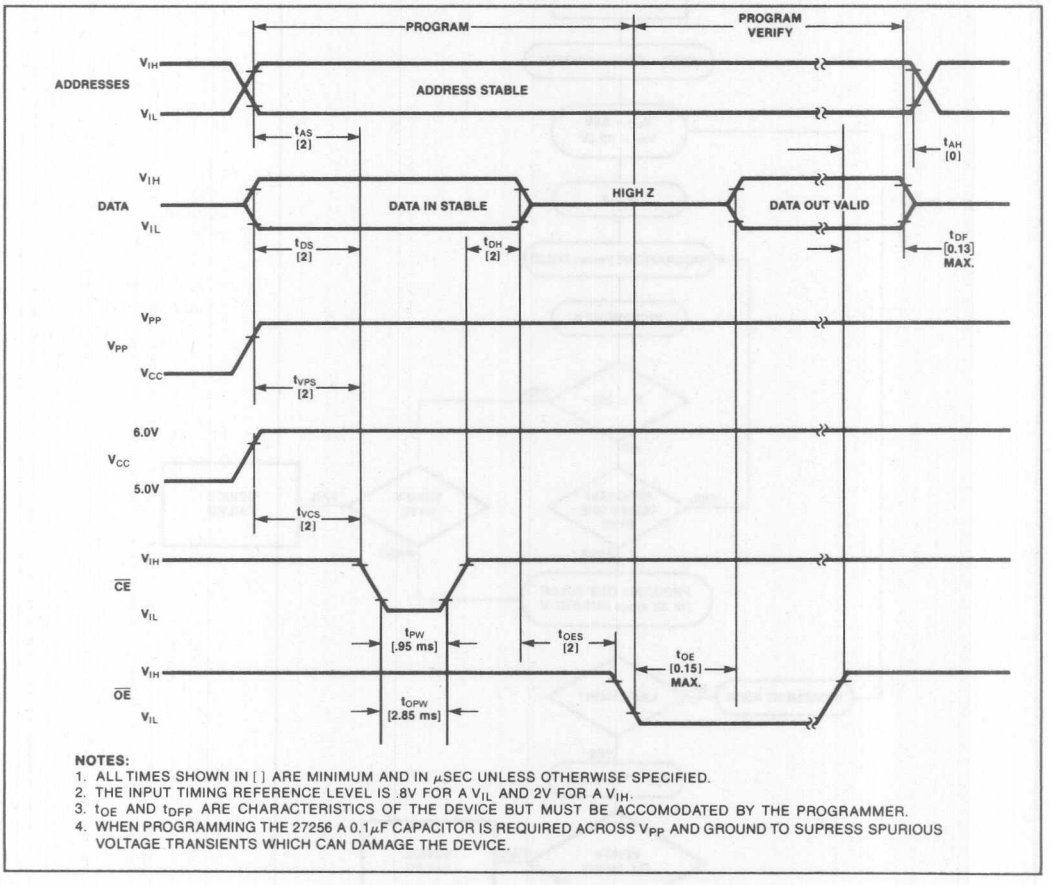


Figure 3. int$_e$ligent Programming™ Waveforms

Table 1. 27256 Operational Modes

	A9	$\overline{CE}$	$\overline{OE}$	V_{PP}	I/O
Read	X	V_{IL}	V_{IL}	V_{CC}	Data Out
Int$_e$ligent Programming™	X	V_{IL}	V_{IH}	V_{PP}	Data In
Verify	X	V_{IH}	V_{IL}	V_{PP}	Data Out
Program Inhibit	X	V_{IH}	V_{IH}	V_{PP}	High Z
Output Disable	X	V_{IL}	V_{IH}	V_{CC}	High Z
Standby	X	V_{IH}	X	V_{CC}	High Z
Int$_e$ligent Identifier™	VH	V_{IL}	V_{IL}	V_{CC}	Code

NOTES:
1. X = V_{IH} or V_{IL}
2. VH = 12.0 ± 0.5 Volts

SUMMARY

The introduction of the 27256 continues Intel's leadership in high density EPROM storage capability. With this density increase comes the ability to design in system firmware, creating a more reliable, user-friendly environment for the computer operator. The int$_e$ligent Programming Algorithm developed for Intel's 2764 and 27128 EPROMs, provides many benefits, including lower costs, which result from higher system manufacturing throughput. In line with this strategy, the 27256 int$_e$ligent Programming Algorithm was designed to meet the technology requirements of the new device. With this algorithm, the 27256 may be programmed according to the waveforms shown in figure 3, ensuring programming reliability and efficiency.

intel

ARTICLE
REPRINT

AR-260

March 1983

E-PROMs Graduate To 256-K Density With Scaled N-Channel Process

M. Van Buskirk, M. Holler, G. Korsh,
B. Lee, S. Lee, D. Tang, G. Teng,
S. Fouts, P. Dang, and W. Fisher

MARCH 1983
ORDER NUMBER: 210988-001

E-PROMs graduate to 256-K density with scaled n-channel process

With 32-K bytes per chip, erasable programmable read-only memory can carry application software for business and personal computers

by M. Van Buskirk, M. Holler, G. Korsh, B. Lee, S. Lee,
D. Tang, G. Teng, S. Fouts, P. Dang, and W. Fisher, *Intel Corp., Santa Clara, Calif.*

☐ Since the introduction of the 2-K 1702 in 1971, the erasable programmable read-only memory has shaken off its reputation as a mere prototyping tool and emerged as a major commodity, worth some $240 million last year in the U. S. alone. Central to that growth, a heady pace of process and circuit innovations has doubled E-PROM densities every one to two years.

The advent of the 256-K chip—exemplified by the 27256 from Intel—signals the crossing of key technical hurdles and with it an open path to accelerated development of megabit and larger arrays. Though clearly the child of earlier generations of E-PROMs, the part has been thoroughly scaled down to achieve thinner oxides and minimum features of 1 micrometer. New sensing and decoding circuitry are incorporated as well.

Along the path to even larger arrays, the price per bit of E-PROMs will continue to drop, closing in on that of ROM, the least expensive semiconductor storage. Indeed, the 256-K ROM has only a year's jump on the 27256. The cost of E-PROM is projected at just 50% more than that of ROM in 1985: 6 versus 4 millicents per bit.

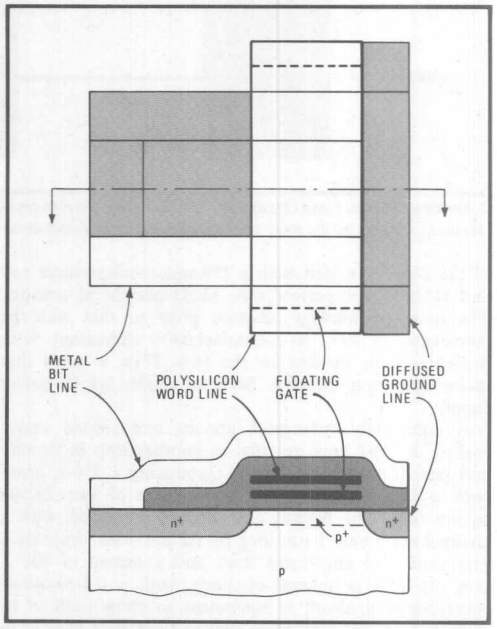
Achieving 256-K density in an E-PROM calls for a host of advances working in concert, dramatic scaling down of device dimensions being only the most obvious one. Processing changes also accompany the smaller geometries and thinner layers. New designs for the decoding and sensing circuitry cope with the worsening problem of statistical variations in device parameters among the more than quarter million bits in the array.

The flexibility of alterable nonvolatile storage coupled with the economy of high chip density promises intriguing possibilities for the architect of microsystems, particularly portable computers for the mass market. As the table on page 93 suggests, significant system and application software fits in a very few 256-K arrays. For example, two or three chips can carry a Pascal compiler and a sophisticated word-processing program.

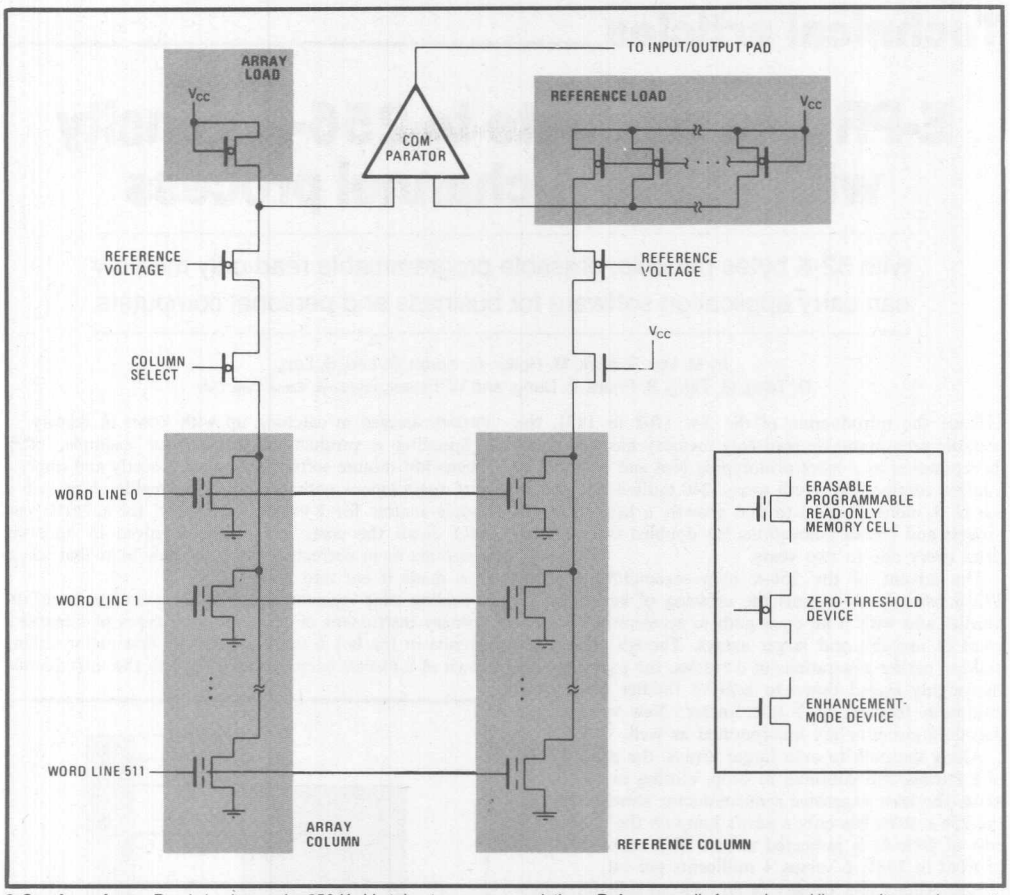
To the user of a machine incorporating such firmware, the friendliness and convenience of fast, 200-nanosecond memory accessed with the push of a button contrasts sharply with the fuss and delay of floppy disks. Further, for the end user and equipment maker alike, E-PROM continues to offer compelling advantages over ROM, advantages that will come at an ever lower premium as E-PROMs succeed in catching up with ROMs in density.

Speeding a product to market, for example, often means last-minute software changes, a costly and impractical requirement with mask-programmable chips, but a simple matter for E-PROMs. Similarly, the E-PROM can hold down the costs and delays involved in updating programs or in correcting those last few bugs that somehow made it out into the field.

Scaling chip geometries down in size has been the primary instrument of progress in all types of integrated circuits in the last 6 to 10 years. The first major scaling down of E-PROMs occurred in 1980 with the introduction



1. Scaled. Two-micrometer design rules squeeze the E-PROM cell down to 6 by 6 μm. The active channel area, beneath the floating polysilicon gate, is just 1 by 1.2 μm. The n^+ regions are 0.5 μm deep.



2. Sensing scheme. Read circuitry on the 256-K chip tolerates process variations. Reference cells for each word line monitor read currents; identical geometry for the array and reference load transistors ensures a constant current ratio for sensing.

of the 2764 64-K part with a 159-square-micrometer cell and H-MOS (high-performance MOS) peripheral circuits. The major technology advance prior to that was the conversion in 1977 to n-channel MOS technology with depletion-mode devices for the 16-K 2716, a move that made operation possible from a single 5-volt power supply.

A completely redesigned process and second major scaling in 1982 have resulted in another leap in density and performance for E-PROMs, producing a 256-K array with a 6-by-6-μm cell, for a chip size of just 28,500 square mils. The floating-gate storage transistor, with a channel effectively 1 μm long by 1.2 μm wide, employs a first oxide 325 angstroms thick and a second of 400 Å (Fig. 1). The peripheral circuitry, with a 0.5-picojoule speed-power product, is equivalent to chips built in H-MOS II, a second-generation high-performance MOS technology that uses 2-m channel lengths and 400-Å gate oxides for minimum gate delays of 0.4 ns.

The practical constraint of maintaining the same 5-V power supply used in previous technologies while scaling transistor sizes stresses the materials constituting the devices: both substrate and gate dielectrics are exposed to much higher electric fields. The fabrication process must build a margin of safety into the chip to prevent unwanted effects like time-dependent oxide failure, pn-junction breakdown, and parasitic MOS-transistor action between adjacent diffused regions.

In E-PROMs, the high voltage required for programming the cells further aggravates the situation. In fact, in the storage transistor, the oxide surrounding the floating gate must be flawless, or else electrons leak off, losing the stored data. For that reason, the programming voltage in the 27256 is scaled down to 12.5v (see "Scaling down the E-PROM cell," opposite). With that scaling, the 400-Å oxides in the transistors of the peripheral circuitry that route the programming voltage to the array experience an electric field of 3.25 megavolts per centimeter, rough-

Scaling down the E-PROM cell

A two-step scaling-down process starting from the 64-K erasable programmable read-only memory (the 2764) has resulted in the 256-K chip designated the 27256 by Intel. The original 64-K E-PROM represented a modest scaling of the 16-K, using positive photoresists and projection-printer lithography. The first step beyond the 64-K chip called for wafer-stepping lithography for levels such as the first polysilicon and contact openings, where significant area could be saved without redesigning the devices or changing the other process steps. Those moves substantially reduced the size of the 64-K chip and ushered in a 128-K version.

At the same time, however, a program was under way to scale down all the design rules and use the stepper lithography to full advantage. That complete redesign of the process not only produced the 256-K array, but suggests that, from a device design perspective, 512-K and 1-megabit arrays will be feasible in the next two to three years. However, substantially better control over dimensions will have to accompany that continued scaling. Several variations were made upon the so-called classical scaling theory and can be understood with reference to the operation of the E-PROM cell.

The E-PROM cell is a simple modification of a conventional n-channel MOS enhancement-mode transistor whose drain connects to a bit line and whose source is grounded. The transistor's gate floats and is controlled by capacitive coupling to a polysilicon word line overlying the gate. Current conducted through the transistor is read as a logical 1, and the absence of current as a 0.

The cell is programmed to preserve a nonconducting state by applying high voltages to the word and bit lines simultaneously. Under those conditions, hot electrons are injected from the channel to the floating gate, charging it to a negative voltage. With the gate charged by the trapped electrons, the word line cannot couple enough voltage to it to turn on the transistor during a subsequent read operation. Exposing the cell to ultraviolet light elevates the trapped electrons' energy to the level of the conduction band of the surrounding oxide, and their mutual repulsion then causes them to flow off the gate.

The cell for the 256-K chip was derived from that of its 64-K predecessor using a scaling factor of 2. As shown in the table, the constant-field scaling theory was followed almost exactly for the device dimensions, oxide thicknesses, and programming voltage. Significant departures from the formulas were made in the read voltage, which was not scaled at all, and the channel-doping concentration, which was more than doubled (in fact almost quintupled). The doping was boosted this much to increase the electric fields in the channel and thereby improve the programming efficiency. Such an increase was feasible because the read voltage was not scaled: the threshold voltage is set as a fraction of the read voltage to ensure the optimal logic threshold, and it therefore could also remain unchanged.

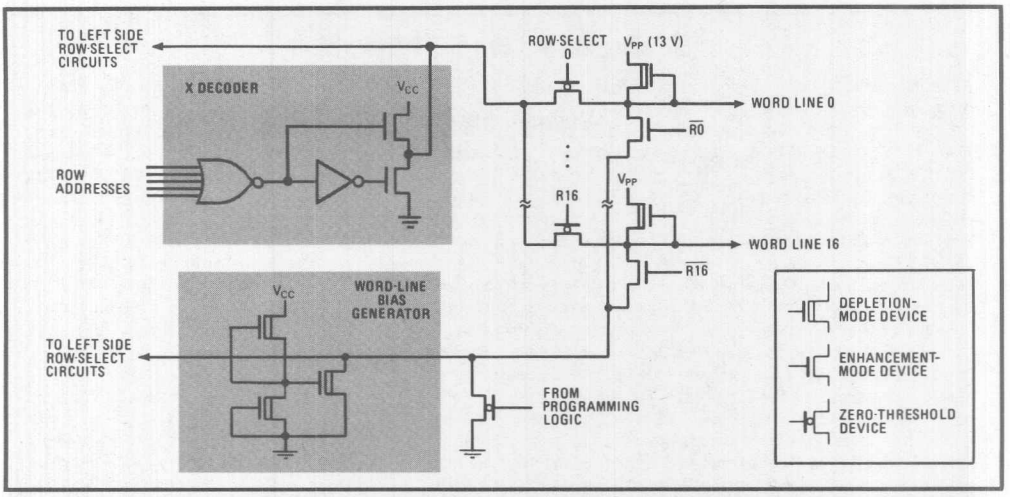
In the scaling theory, the read voltage would be scaled to prevent an increase in the electric field—and hence the stress—across the oxides. In an E-PROM, however, special treatment is required for the oxides, which see a worst-case field due to the high programming voltage. Thus, with the programming voltage scaled and oxides with the requisite integrity fabricated, the read voltage presents no problem. In fact, leaving the read voltage at 5 volts instead of scaling it to 2.5 V substantially increases the read current and speeds up access to the array.

Because of the high channel fields caused by the high dopant concentration, a 5-V drain voltage could suffice to inject hot electrons to the gate during a read, an effect appropriately known as parasitic programming. Therefore, the drain voltage was scaled down well below 5 V.

The devices in the peripheral circuitry were scaled from an effective length of 3.4 micrometers to 1.7 μm, and their oxides were thinned from 650 angstroms to 400 Å, which is equivalent to H-MOS II transistors. Those two changes increased the current of a minimum-length transistor by a factor of only 2.3: the increase in channel doping of the enhancement-mode devices degrades the electron mobility, limiting the current increase. Because the supply voltage was not scaled, the speed-power product of the technology is greater than it otherwise might have been, decreasing only by half to 0.5 picojoule.

SCALING DATA FOR THE 256-K ERASABLE PROGRAMMABLE READ-ONLY MEMORY

Parameter	64-K chip	Constant-field scaling formula	Theoretical 256-K chip	Actual scaling formula	Actual 256-K chip
Cell area	159 μm^2	K^{-2}	40 μm^2	$\approx K^{-2}$	36 μm^2
Floating-gate oxide	725 Å	K^{-1}	363 Å	$\approx K^{-1}$	325 Å
Control-gate oxide	900 Å	K^{-1}	450 Å	K^{-1}	450 Å
Channel doping	2×10^{16} cm^{-3}	K	4×10^{16} cm^{-3}	$K^{+2.2}$	9×10^{16} cm^{-3}
Threshold voltage	1.6 V	K^{-1}	0.8 V	1	1.6 V
Cell current	55 μA	K^{-1}	28 μA	$K^{+0.5}$	80 μA
Read voltage	5 V	K^{-1}	2.5 V	1	5 V
Program voltage	21 V	K^{-1}	11 V	$\sim K^{-0.7}$	12.5 V



3. Biased decoder. With a 0.5-volt bias applied to deselected word lines during a read operation, depletion-mode pass transistors can be used in the decoder. They increase the selected word line's voltage, speeding access and extending the allowed power-supply range.

ly equal to the electric field found in previous parts.

The thin oxide between the substrate and floating gate illustrates the engineering that went into each of the process steps. Early in the development, that 325-Å-thick oxide showed defect densities of around 100/cm². (The oxide integrity is judged with a sensitive measurement of the dc current flowing through a large-area capacitor biased to 15 V.) Based on the total active area, A, of the channels of the 256-K chip's array transistors, a simple model for the yield, such as e^{-AD}, predicts the loss due to that defect density, D, at 54%, an unacceptably high figure for a single process step.

With careful adjustment of process parameters, the thin-oxide defect density was reduced below 5/cm², raising the yield of that step to an estimated 96%. Similarly, each new step could be fine-tuned independently of any other procedures, simplifying the process debugging. Despite the many new procedures, the overall structure and the sequence of steps do not depart radically from previous E-PROMs, a help in the retraining of manufacturing personnel.

In particular, the shared drain contact, common source diffusion, and self-aligned floating polysilicon gate of the E-PROM cell have not changed conceptually since the 2716 of five years ago. What has changed dramatically is the minimum feature size: the evolution from projection printing and wet etching to wafer-stepping lithography and anisotropic plasma etching on critical levels has shrunk the 27256's cell to the size of a contact hole on the old 2716.

Most elements of the E-PROM process are affected by that radical shrink. The field oxide that isolates adjacent diffusions is thinned to 0.6 μm, a move that cuts the length of the bird's beak in half. (The bird's beak, or transition region, between the oxide and the active device is wasted area.) As a result, the minimum spacing on the mask between adjacent field oxide regions is only 2 μm. The thinner oxide is possible thanks to the lower programming voltage, which reduces the chance of parasitic transistor action beneath the oxide.

The dose of boron implanted in the transistor channels to set their threshold voltage is increased, compensating for the effects of the shorter, narrower channels. In turn, the gate oxides are thinned to boost the transconductance of the more highly doped channels. The arsenic for the source and drain regions is implanted 0.5 μm or shallower to control the channel length and forestall punchthrough. To prevent the metalization from penetrating those shallow junctions, an aluminum-silicon alloy substitutes for the usual aluminum.

Although process development goes far to ensure a reliable part with good margins, new circuits were also required for the E-PROM's jump to 256-K density. For one, some statistical variation in cell characteristics is inevitable, and the more bits per chip, the greater the likelihood that one or more bits will fall outside the acceptable limits. Thus, circuits that compensate for these process-induced variations in effect raise yield. Sensing circuitry offers one example.

The 27256 uses two entire columns of reference cells, associating two cells with each of the 512 word lines rather than one with each of the eight comparators in the output section (Fig. 2). The tolerance to variations in read currents quintuples that in the 2764. Undoubtedly, future E-PROMs will extend this concept again, incorporating multiple reference columns.

Before the 2764, E-PROMs employed single-ended sense amplifiers, which are comparators that switch when the input current exceeds some threshold, called the trip current, which is not a function of any cell parameters. To make the trip current dependent on the read current itself, the 2764 used a differential comparator, which compares the read current with a fixed fraction of the current from a reference cell—a replica of

the array cells. If a cell's read current is lower than expected, then the reference cell's current will probably be low as well, adjusting the trip current accordingly. Such a circuit not only continues to operate under process variations, but also allows the design of a comparator optimized for speed.

Unfortunately, carrying that scheme even further on the 27256 and increasing the number of reference cells naturally leads to a wider distribution of their currents, and the conventional differential comparator could introduce an error because of that variation. To compensate for that error, a constant-current–ratio differential comparator was introduced. In it, the lower impedance reference load is obtained by connecting several transistors in parallel, each of which is an identical copy of the array load device (see Fig. 2 again). The parallel connection then gives the correct impedance without changing the transistor operating point, permitting correct operation over a much wider range of read current.

Extending a scheme

In the usual comparator, the dimensions of the load device on the reference cell or column are adjusted to give a lower impedance than that of the load device on the array column. The voltage drops across the different impedances are the differential input to the comparator. However, the different load-device dimensions change the devices' threshold voltages, putting the reference load at a different operating point. That difference means that the ratio of the array current to the reference current changes with variations in the reference current.

Another innovation, in the row-decoder circuits, helps speed access time and extend the power-supply variations the chip will tolerate. The scheme devised for the 27256 uses a negative-threshold pass transistor; however, it also incorporates a bias-voltage generator to hold the deselected word lines at about 0.5 V, rather than 0 V, during a read operation (Fig. 3). The bias current for the generator is supplied by pull-up transistors connected to the programming voltage supply. The 0.5-V shift suffices to reduce the leakage current without turning on deselected cells during a read operation.

However, during programming, when the high voltage applied to selected columns can couple to the floating gates, the 0.5-V bias would be enough to partially turn on deselected cells. Thus, a shunt transistor is included in the bias generator to pull the deselected word lines all the way to ground during programming.

In a conventional decoder, if a positive-threshold device is used, the word-line voltage rises very slowly above the level of one threshold below the supply voltage, slowing access to the cells. It also ups the minimum power-supply voltage by about 1 V. (The minimum supply voltage is the cell's threshold voltage, plus that voltage required to generate a detectable current difference between programmed and erased cells, typically about 3 V.)

Using a negative-threshold device overcomes both those effects, but introduces its own problems if not accompanied by a bias-voltage generator. With a depletion-mode pass transistor, the deselected devices still pass a significant leakage current. That current pulls down the decoder output below the supply voltage so that the selected word line still receives a reduced voltage.

TYPICAL PROGRAM SIZES FOR 256-K MEMORIES

Program	Bytes	Number of 256-K erasable programmable read-only memories
Basic interpreter	20-K to 32-K	5/8 to 1
Pascal compiler	32-K to 40-K	1 to 1 1/4
Asteroids	8-K	1/4
Screen-oriented editor	10-K	1/3
Word processor	32-K to 48-K	1 to 1 1/2
Spelling dictionary	12-K	3/8
Relational data base	32-K to 128-K	1 to 4

The 27256 also illustrates the utility of two circuit features trademarked as the intelligent Identifier and the intelligent Programming Algorithm. A persistent problem for the user of scaled-down E-PROM technology is the changing programming requirements. Each new generation forces customers to convert to lower voltages and altered algorithms. The identifier, an on-chip, unalterable, 2-byte code, specifies the chip's manufacturer, programming algorithm, voltage, and pulse width.

Saving progamming time

With ever denser E-PROM arrays, the programming algorithm proves its worth in saving programming time. This adaptive, closed-loop algorithm varies the width and the number of program pulses to reach an adequate margin in the minimum time, typically 4 milliseconds per byte for the 27256.

The old programming algorithm, developed for the 2716, used a fixed 50-ms pulse width. That width is determined by the worst-case programming time, that is, the longest time any bit in the array will need for complete programming. As with other device characteristics, process variations lead to a distribution of programming times; the fixed-width pulse must be long enough to program the slowest bit in the array. At the 256-K level, chances are good that at least one bit would need a very long programming pulse.

The closed-loop programming algorithm requires a method of determining the programmed cell "margin," or retention characteristics. Margin has generally been represented as the maximum supply voltage at which both programmed and erased cells can be read. A programmed cell is one in which the threshold voltage has been forced to a higher voltage. Increasing the supply voltage boosts the voltage applied to a cell's gate; eventually, the higher threshold is reached, turning on a cell that is intended to remain off.

Thus, the strategy in the programming algorithm is to verify that a cell is programmed at an elevated supply voltage of 6 V. The procedure begins by applying a 1-ms pulse to the first byte. At the end of the pulse, the chip tries to read the programmed cells. If they are programmed, an additional 3-ms pulse is applied, boosting the margin about 1.5 V. If the cell is not programmed, 1-ms pulses continue to be applied until programming is verified. When it is, a pulse three times as long as the sum of the 1-ms pulses already applied adds the required margin. To program a 27256 takes about 3 minutes. □

March 1983

Versatile Algorithm, Equipment Cut EPROM Programming Time

Don Knowlton
EPROM Product Line
Manager
Intel Corp.

ORDER NUMBER: 230614-001

Versatile algorithm, equipment cut EPROM programming time

Programming high-density EPROMs for large-volume applications can entail significant time and expense. To minimize both, use programming software and hardware that recognize the different characteristics of individual EPROM cells.

Don Knowlton, Intel Corp

Using the capabilities of μP-based EPROM programming equipment and employing a new algorithm that recognizes differences among EPROM cells can dramatically reduce programming time for the newest high-density devices. The typical factor-of-six reduction this system can achieve results in considerable cost savings for large-volume applications. The time required to program an 8k-byte 2764, for example, drops from 7 min to an average of 1.25 min with the procedure. As a bonus, the technique helps ensure that the device receives adequate programming—in terms of memory-cell charge—to maintain long-term reliability.

Reducing programming time and costs for EPROMs has become increasingly important because the chips have become a cost-effective, easy-to-use alternative to masked ROM in high-volume applications requiring code flexibility or simplified inventory—a major switch from EPROMs' original small-volume prototyping applications. And volume usage makes EPROM programming a significant manufacturing consideration, subject to minimization efforts.

TABLE 1— EPROM PROGRAMMING-TIME EVOLUTION

DEVICE	BYTES	PROGRAMMING TIME (MIN)
1702A	256	2.0
2708	1024	1.5
2716	2048	1.75
2732	4096	3.5
2764	8192	7.0
2764*	8192	1.25
27128*	16384	2.5

*USING ENHANCED PROGRAMMING ALGORITHM DESCRIBED IN THE TEXT.

Higher densities increase programming time

Despite dramatic (eightfold) increases in EPROM storage capacity, early technology improvements (from p-channel MOS through 3-power-supply NMOS to single-power-supply NMOS) held EPROM programming times to less than 2 min (**Table 1**). Since the appearance of the 16k-bit EPROM, however, the minimum time for reliably programming each EPROM cell has remained constant at 45 msec, doubling the total device programming time with each doubling of density. The result is an increase in the per-unit cost of programming—involving either additional time (labor) or extra equipment.

The programming procedure currently in use for most EPROMs isn't yet a major burden on most users, though; it uses a nominal 50-msec pulse per EPROM byte, resulting in a total programming time of approximately 1.5 min for a 16k-bit chip. With the introduction of the 2764 (64k bits) and devices with even higher density, however, programming times have increased. A 256k-bit EPROM, for example, would require 24 min for programming by this conventional method.

Most EPROM cells program in much less than 45 msec, however. In fact, empirical data (**Fig 1**) shows that very few cells require longer than 8 msec for programming. Therefore, a procedure that takes into account the characteristics of individual EPROM cells can significantly reduce a device's programming time.

Arbitrarily reducing programming time is risky, though, because a cell's ability to achieve and maintain its programmed state is a function of this time (see **box,** "EPROM programming"). What's needed, therefore, is a way to verify the level to which individual cells have been programmed.

Fortunately, such techniques exist. By determining the charge stored in a cell relative to the minimum charge needed to program the cell to a detectable level, you can check for a program margin that assures reliable EPROM operation.

One way to check program margin involves varying the select-gate voltage, V_{CC}. And although you wouldn't

Reduce EPROM programming time with an intelligent algorithm

necessarily use this procedure in the actual programming of an EPROM, it serves to illustrate some EPROM characteristics and thus merits discussion.

An erased cell (having a ONE output) with no charge on its floating gate has a characteristic similar to an ordinary NMOS I-V curve **(Fig 2).** As you begin programming the cell (to a ZERO), the cell threshold as a function of select-gate voltage begins to increase. Thus, if you externally increase V_{CC}, connected to the select gate, you can determine cell threshold by observing the V_{CC} value at which a ZERO-to-ONE transition occurs on the output. If a transition occurs within V_{CC}'s normal operating range of 4.75 to 5.25V, the EPROM cell is inadequately programmed.

Some examples illustrate the process. First, consider a cell programmed for time t_1-t_0, storing a charge Q_1 on the floating gate:

$$Q_1 = \int_{t_0}^{t_1} i(t)\,dt.$$

If the charge isn't adequate to fully program the cell, the EPROM output exhibits a ZERO-to-ONE transition as V_{CC} increases through a level less than the 5.25V max operating voltage.

Now consider a second cell programmed for time t_2-t_0, storing a charge Q_2 on the floating gate. If increasing V_{CC} causes a ZERO-to-ONE transition at exactly 5.25V, the cell has exactly the amount of charge required for programming. However, the cell has no margin for tolerating charge loss or small variations in V_{CC} beyond the specification maximum, and such a device might not be reliable in long-term operation.

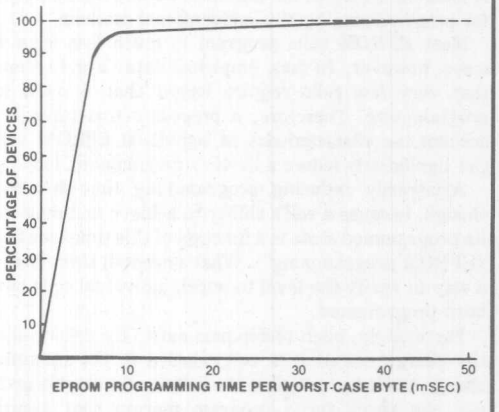


Fig 1—Most EPROM cells program in 8 msec or less, *but conventional programming algorithms needlessly use much more time.*

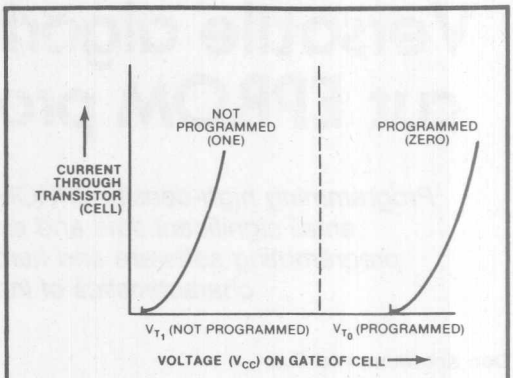


Fig 2—An erased EPROM cell *has an ordinary NMOS I-V curve (left), while a fully programmed cell exhibits a modified curve (right). The curve of an inadequately programmed cell lies between the two; such a cell can exhibit both ONE and ZERO outputs within V_{CC}'s normal operating range.*

Finally, consider a device programmed for time t_2+T-t_0. In this case, charge added by the additional programming time T increases programming margin. The ZERO-to-ONE transition detected while increasing V_{CC} thus occurs when V_{CC} is greater than 5.25V, providing assurance of reliability.

Margin checking doesn't occur in conventional EPROM programming, however. Instead, each EPROM cell receives a 45- to 55-msec write pulse, and manufacturers ensure program margin by screening out devices having bytes that don't program within 45 msec. This programming procedure is thus an open loop—no actual verification of margin occurs.

Improved algorithm uses closed-loop technique

In contrast, the trademarked inteligent Programming algorithm, a procedure devised by Intel for programming high-density EPROMs, guarantees reliability through the closed-loop technique of margin checking. It ensures that an EPROM cell's intended ZERO output won't become a ONE within V_{CC}'s normal operating range.

The algorithm begins by setting V_{CC} to 6V (higher than required for normal operation but necessary to provide programming margin), setting the programming voltage V_{PP} to 21V and then iteratively supplying 1-msec LOW-going programming pulses to the EPROM's program-disable pin $\overline{PGM}$. After each pulse, the algorithm checks the EPROM's output for the desired programmed value. If the output is incorrect, the algorithm repeats the pulse-and-check operation; incorrect output after 15 pulses causes rejection of the EPROM device.

If the EPROM is fully functional, however, one of the pulses results in proper EPROM output. At that point, the algorithm supplies still another programming pulse—this one four times longer than the combined length of the previously applied 1-msec pulses. This longer pulse helps ensure that the EPROM cell has adequate programming margin for reliable operation. Although the pulse can be as long as 60 msec, very few EPROM bytes require this much programming time. In fact, most EPROM bytes program with only one or two 1-msec pulses, so a typical total programming time per byte is 5 to 10 msec.

TABLE 2—
EPROM PROGRAMMER DESIGN RANGES

PARAMETER	MINIMUM	MAXIMUM
V_{CC}	4.5V	7.0V
V_{PP}	10.0V	25.0V
t_{PW}*	0.1 mSEC	60 mSEC

*PROGRAMMING-PULSE WIDTH

Use commercial programmers or design your own

Most commercial EPROM programming equipment can accommodate this programming algorithm with minor changes to hardware and software. For example, Data I/O Models 120A and 121A programmers require that firm's Revision D software; the company's Unipak, Unipak II and Mospak programmers require Revisions 004, 001 and 003, respectively. You need Revision B software from Pro-Log to use the algorithm on that firm's M980 control unit; you also need the PM9080 module and PA28-80 socket adapter.

If you design your own equipment to implement the algorithm, you must consider several application factors in weighing performance and flexibility against cost. For instance, if your application is dedicated to programming only the 2764, you can construct a system totally in hardware to minimize costs. You need only three fixed-voltage power supplies (5, 6 and 21V), and you can generate pulses with simple devices such as one-shots. Logic arrays or a μP can perform the counting and algorithm sequencing.

Systems that program several types of EPROMs require a more general approach; they're best implemented with a μP and programmable power supplies. Look-up tables in the program-store memory can access algorithms for various EPROMs in this type of system, and tables can contain pulse widths and voltage parameters. Most parameters are common to many EPROMs, so programming the various types is easy.

Moreover, you can employ software in such systems to count the 1-msec pulses and construct various other pulse widths to satisfy the programming algorithm's timing requirements.

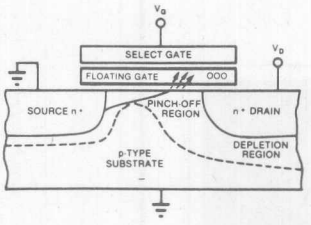
Supply-voltage generation, however, proves more difficult. An elegant and flexible solution involves using software-controlled programmable-output power supplies to deliver V_{PP} and V_{CC}, thus providing compatibility with future designs.

As an alternative, you could employ selectable supplies. Unfortunately, this less costly approach might

EPROM programming

Information storage in an EPROM results from electrically charging a floating gate in a stacked-gate MOS transistor. To charge the gate, you raise both the select-gate voltage V_G and the drain voltage V_D to a high positive level while holding source and substrate at ground potential **(figure);** suitable drain and gate voltages cause the transistor to operate in saturation.

The acceleration of electrons in the transistor's pinch-off region results from the high electric field. Some of these electrons acquire enough energy to enter the conduction band of the silicon dioxide (which electrically isolates the floating gate under normal operating voltages) and get attracted to the positive potential of the floating gate.



Information storage in an EPROM cell *results from the transfer of charge to an MOS transistor's floating gate during the programming operation.*

The basic equation describing the behavior of cell programming is:

$$i = C_{FG}\frac{dV}{dt},$$

where C_{FG} is the floating-gate capacitance. The charge accumulated on the floating gate is therefore:

$$Q = \int_{t_0}^{t_1} i(t)\,dt = C_{FG}V.$$

Thus, the amount of charge stored in the cell and the resultant level of programming are functions of the cell programming time $t_1 - t_0$.

Get a factor-of-six speedup with individual-cell programming

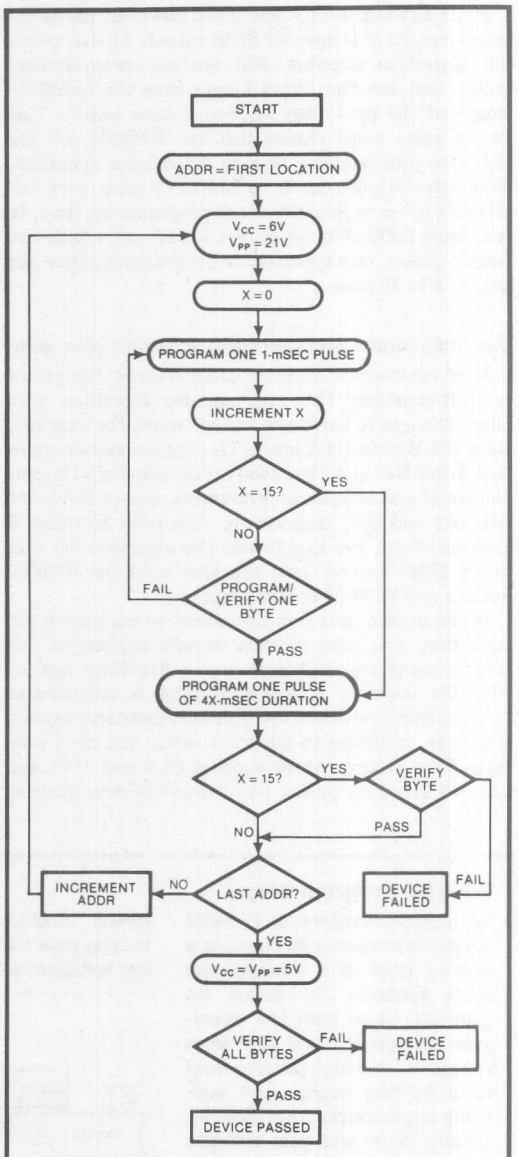
leave the equipment lacking the voltage required for programming future devices. **Table 2,** however, shows suggested voltage and timing ranges for equipment design that should meet anticipated requirements for Intel MOS EPROMs.

Equipment designed to program several EPROMs in parallel requires special design considerations. Devices that fail to program slow down the entire programming operation if each address is allowed to program for the maximum time (pulse counter=15). A ganged unit, however, should detect programming failures immediately and disable faulty devices to minimize programming time for the other EPROMs. This step is particularly critical in high-volume programming.

Good design practices help

Furthermore, following many of the design rules used in dynamic-RAM applications can help you improve the reliability of high-density MOS-EPROM programmers. For example,

- Provide well-regulated power supplies at the EPROM. Short, low-inductance traces, high-frequency capacitive decoupling and ground-plane routing all reduce the effects of current surges and are necessary for high speed.
- Make sure that all signals are clean. Minimize undershoot, overshoot and glitches. Use an oscilloscope with at least a 150-MHz bandwidth to find potential problems.
- Observe absolute maximum specs on V_{PP}. Because V_{PP} is the highest voltage applied to the chip, it's usually the closest voltage to an EPROM's physical limits. Even small glitches exceeding absolute maximum ratings can result in chip destruction, so use a high-bandwidth oscilloscope to look for overshoot when setting V_{PP}. EDN



Intelligent programming algorithms, *like the one for Intel's 27128 described by this flowchart, cut programming time by more than 80%.*

Author's biography

Don Knowlton is EPROM product-line manager at Intel's Nonvolatile Memory Div (Santa Clara, CA), where he's responsible for EPROM strategic marketing. Before joining Intel, he worked at Advanced Micro Devices. Don holds BSEE and MSEE degrees, both from Purdue University. In his spare time, he enjoys playing 5-string banjo.

2764A
ADVANCED 64K (8x8) UV ERASABLE PROM

- **200 nsec Standard Access Time —HMOS II*-E Technology**
- **Low Power —75 mA Maximum Active —35 mA Maximum Standby**
- **Two Line Control**
- **int$_e$ligent Programming™ Algorithm —Fastest EPROM Programming**
- **int$_e$ligent Identifier™ Mode —Automated Programming Operations**
- **Compatible with 2764, 27128, 27256**
- **±10% V_{CC} Tolerance Available**

The Intel 2764A is a 5V only, 65,536-bit ultraviolet erasable and electrically programmable read-only memory (EPROM). The 2764A is an advanced version of the 2764 and is fabricated with Intel's HMOSII-E technology which significantly reduces die size and greatly improves the device's performance, power consumption, reliability and producibility.

The standard 2764A access time is 200 ns which is an improvement over the 2764 standard access time of 250 ns. This is compatible with high-performance microprocessors, such as Intel's 8 MHz iAPX 186 allowing full speed operation without the addition of WAIT states. The 2764A is also directly compatible with the 12 MHz 8051 family.

Several advanced features have been designed into the 2764A that allow fast and reliable programming—the int$_e$ligent Programming Algorithm and the int$_e$ligent Identifier Mode. Programming equipment that takes advantage of these innovations will electronically identify the 2764A and then rapidly program it using an efficient programming method.

The 2764A also offers reduced power consumption compared to the 2764. The maximum active current is 75 mA while the maximum standby current is only 35 mA. The standby mode lowers power consumption without increasing access time.

Two-line control and JEDEC-approved, 28 pin packaging are standard features of all Intel higher density EPROMs. This ensures easy microprocessor interfacing and minimum design efforts when upgrading, adding or choosing between non-volatile memory alternatives.

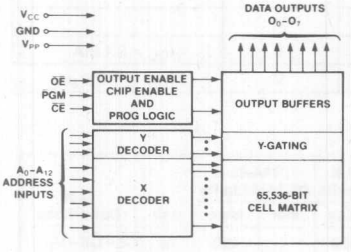
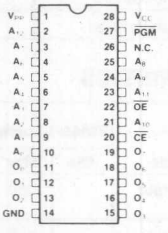


Figure 1. Block Diagram

27256	27128	2732A	2716
V_{PP}	V_{PP}		
A_{12}	A_{12}		
A_7	A_7	A_7	A_7
A_6	A_6	A_6	A_6
A_5	A_5	A_5	A_5
A_4	A_4	A_4	A_4
A_3	A_3	A_3	A_3
A_2	A_2	A_2	A_2
A_1	A_1	A_1	A_1
A_0	A_0	A_0	A_0
O_0	O_0	O_0	O_0
O_1	O_1	O_1	O_1
O_2	O_2	O_2	O_2
Gnd	Gnd	Gnd	Gnd

2764A



2716	2732A	27128	27256
		V_{CC}	V_{CC}
		$\overline{PGM}$	A_{14}
V_{CC}	V_{CC}	A_{13}	A_{13}
A_8	A_8	A_8	A_8
A_9	A_9	A_9	A_9
V_{PP}	A_{11}	A_{11}	A_{11}
$\overline{OE}$	$\overline{OE}$ V_{PP}	$\overline{OE}$	$\overline{OE}$
A_{10}	A_{10}	A_{10}	A_{10}
$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
O_7	O_7	O_7	O_7
O_6	O_6	O_6	O_6
O_5	O_5	O_5	O_5
O_4	O_4	O_4	O_4
O_3	O_3	O_3	O_3

NOTE: INTEL "UNIVERSAL SITE"-COMPATIBLE EPROM PIN CONFIGURATIONS ARE SHOWN IN THE BLOCKS ADJACENT TO THE 2764A PINS

Figure 2. Pin Configurations

MODE SELECTION

MODE \ PINS	$\overline{CE}$ (20)	$\overline{OE}$ (22)	$\overline{PGM}$ (27)	A_9 (24)	V_{PP} (1)	V_{CC} (28)	Outputs (11-13, 15-19)
Read	V_{IL}	V_{IL}	V_{IH}	X	V_{CC}	V_{CC}	D_{OUT}
Output Disable	V_{IL}	V_{IH}	V_{IH}	X	V_{CC}	V_{CC}	High Z
Standby	V_{IH}	X	X	X	V_{CC}	V_{CC}	High Z
Verify	V_{IL}	V_{IL}	V_{IH}	X	V_{PP}	V_{CC}	D_{OUT}
Program Inhibit	V_{IH}	X	X	X	V_{PP}	V_{CC}	High Z
int$_e$ligent Identifier	V_{IL}	V_{IL}	V_{IH}	V_H	V_{CC}	V_{CC}	Code
int$_e$ligent Programming	V_{IL}	V_{IH}	V_{IL}	X	V_{PP}	V_{CC}	D_{IN}

1. X can be V_{IH} or V_{IL}
2. V_H = 12.0V ± 0.5V

PIN NAMES

A_0-A_{12}	ADDRESSES
$\overline{CE}$	CHIP ENABLE
$\overline{OE}$	OUTPUT ENABLE
O_0-O_7	OUTPUTS
$\overline{PGM}$	PROGRAM
N.C.	NO CONNECT

*HMOS is a patented process of Intel Corporation

JUNE 1983
ORDER NUMBER: 230702-002

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias ..... −10°C to +80°C
Storage Temperature ..... −65°C to +125°C
All Input or Output Voltages with Respect to Ground ..... +6.25V to −0.6V
Voltage on Pin 24 with Respect to Ground ..... +13.5V to −0.6V
V_{PP} Supply Voltage with Respect to Ground During Programming ..... +13V to −0.6V

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. AND A.C. OPERATING CONDITIONS DURING READ

	2764A-1, 2764A-2, 2764A 2764A-3, 2764A-4	2764A-20, 2764A-25 2764A-30, 2764A-45
Operating Temperature Range	0°–70°C	0°–70°C
V_{CC} Power Supply[1,2]	5V ±5%	5V ±10%
V_{PP} Voltage[2]	$V_{PP} = V_{CC}$	$V_{PP} = V_{CC}$

READ OPERATION

D.C. CHARACTERISTICS

Symbol	Parameter	Limits				Conditions
		Min	Typ[3]	Max	Unit	
I_{LI}	Input Load Current			10	μA	$V_{IN} = 5.5V$
I_{LO}	Output Leakage Current			10	μA	$V_{OUT} = 5.5V$
I_{PP1}[2]	V_{PP} Current Read			5	mA	$V_{PP} = 5.5V$
I_{CC1}[2]	V_{CC} Current Standby			35	mA	$\overline{CE} = V_{IH}$
I_{CC2}[2]	V_{CC} Current Active		45	75	mA	$\overline{CE} = \overline{OE} = V_{IL}$
V_{IL}	Input Low Voltage	−.1		+.8	V	
V_{IH}	Input High Voltage	2.0		$V_{CC}+1$	V	
V_{OL}	Output Low Voltage			.45	V	$I_{OL} = 2.1$ mA
V_{OH}	Output High Voltage	2.4			V	$I_{OH} = -400$ μA

A.C. CHARACTERISTICS

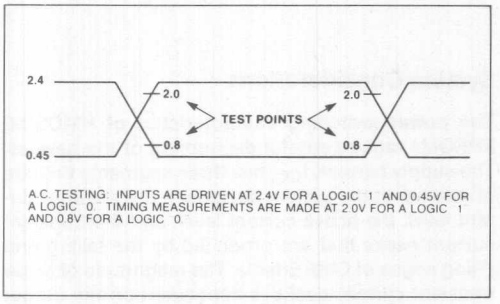
Symbol	Parameter	2764A-1 Limits		2764A-20 & 2764A-2 Limits		2764A-25 & 2764A Limits		2764A-30 & 2764A-3 Limits		2764A-45 & 2764A-4 Limits		Unit	Test Conditions
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max		
t_{ACC}	Address to Output Delay		180		200		250		300		450	ns	$\overline{CE}=\overline{OE}=V_{IL}$
t_{CE}	$\overline{CE}$ to Output Delay		180		200		250		300		450	ns	$\overline{OE}=V_{IL}$
t_{OE}	$\overline{OE}$ to Output Delay		65		75		100		120		150	ns	$\overline{CE}=V_{IL}$
t_{DF}[4]	$\overline{OE}$ High to Output Float	0	55	0	55	0	60	0	105	0	130	ns	$\overline{CE}=V_{IL}$
t_{OH}	Output Hold from Addresses $\overline{CE}$ or $\overline{OE}$ Whichever Occurred First	0		0		0		0		0		ns	$\overline{CE}=\overline{OE}=V_{IL}$

NOTES: 1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. V_{PP} may be connected directly to V_{CC} except during programming. The supply current would then be the sum of I_{CC} and I_{PP1}.
3. Typical values are for $t_A = 25°C$ and nominal supply voltages.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven — see timing diagram on the following page.

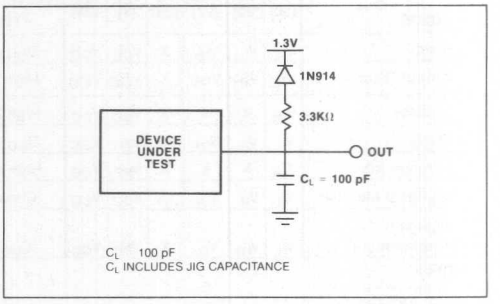
CAPACITANCE ($T_A = 25°C$, $f = 1MHz$)

Symbol	Parameter	Typ.[1]	Max.	Unit	Conditions
C_{IN}[2]	Input Capacitance	4	6	pF	$V_{IN} = 0V$
C_{OUT}	Output Capacitance	8	12	pF	$V_{OUT} = 0V$

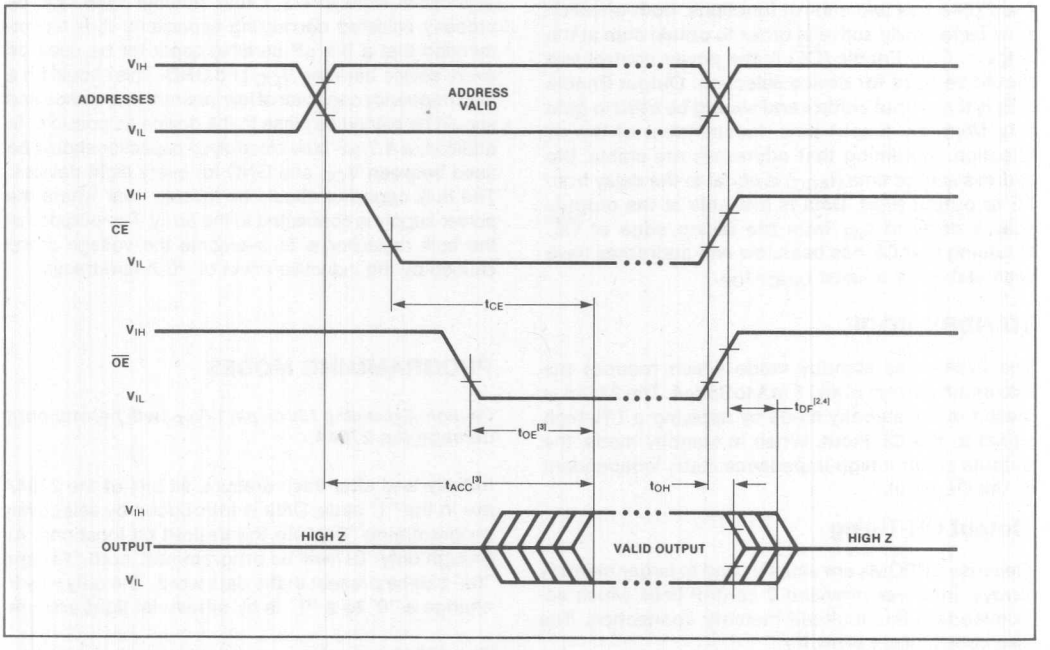
A.C. TESTING INPUT/OUTPUT WAVEFORM



A.C. TESTING LOAD CIRCUIT



A.C. WAVEFORMS



NOTES:
1. Typical values are for T_A = 25°C and nominal supply voltages.
2. This parameter is only sampled and is not 100% tested.
3. $\overline{OE}$ may be delayed up to $t_{ACC} - t_{OE}$ after the falling edge of $\overline{CE}$ without impact on t_{ACC}.
4. t_{DF} is specified from $\overline{OE}$ or $\overline{CE}$, whichever occurs first.

DEVICE OPERATION

The seven modes of operation of the 2764A are listed in Table 1. A single 5V power supply is required in the read mode. All inputs are TTL levels except for V_{PP} and 12V on A9 for int$_e$ligent identifier mode.

Table 1. MODE SELECTION

MODE \ PINS	$\overline{CE}$ (20)	$\overline{OE}$ (22)	$\overline{PGM}$ (27)	A_9 (24)	V_{PP} (1)	V_{CC} (28)	Outputs (11–13, 15–19)
Read	V_{IL}	V_{IL}	V_{IH}	X	V_{CC}	V_{CC}	D_{OUT}
Output Disable	V_{IL}	V_{IH}	V_{IH}	X	V_{CC}	V_{CC}	High Z
Standby	V_{IH}	X	X	X	V_{CC}	V_{CC}	High Z
Verify	V_{IL}	V_{IL}	V_{IH}	X	V_{PP}	V_{CC}	D_{OUT}
Program Inhibit	V_{IH}	X	X	X	V_{PP}	V_{CC}	High Z
int$_e$ligent Identifier	V_{IL}	V_{IL}	V_{IH}	V_H	V_{CC}	V_{CC}	Code
int$_e$ligent Programming	V_{IL}	V_{IH}	V_{IL}	X	V_{PP}	V_{CC}	D_{IN}

NOTES:
1. X can be V_{IH} or V_{IL}
2. $V_H = 12.0V \pm 0.5V$

READ MODE

The 2764A has two control functions, both of which must be logically active in order to obtain data at the outputs. Chip Enable ($\overline{CE}$) is the power control and should be used for device selection. Output Enable ($\overline{OE}$) is the output control and should be used to gate data from the output pins, independent of device selection. Assuming that addresses are stable, the address access time (t_{ACC}) is equal to the delay from $\overline{CE}$ to output (t_{CE}). Data is available at the outputs after a delay of t_{OE} from the falling edge of $\overline{OE}$, assuming that $\overline{CE}$ has been low and addresses have been stable for at least $t_{ACC} - t_{OE}$.

STANDBY MODE

The 2764A has standby mode which reduces the maximum current from 75 mA to 35 mA. The 2764A is placed in the standby mode by applying a TTL-high signal to the $\overline{CE}$ input. When in standby mode, the outputs are in a high impedance state, independent of the $\overline{OE}$ input.

Output OR-Tieing

Because EPROMs are usually used in larger memory arrays, Intel has provided 2 control lines which accommodate this multiple memory connection. The two control lines allow for:

a) the lowest possible memory power dissipation, and
b) complete assurance that output bus contention will not occur.

To use these two control lines most efficiently, $\overline{CE}$ (pin 20) should be decoded and used as the primary device selecting function, while $\overline{OE}$ (pin 22) should be made a common connection to all devices in the array and connected to the $\overline{READ}$ line from the system control bus. This assures that all deselected memory devices are in their low power standby mode and that the output pins are active only when data is desired from a particular memory device.

System Considerations

The power switching characteristics of HMOSII-E EPROMs require careful decoupling of the devices. The supply current, I_{CC}, has three segments that are of interest to the system designer—the standby current level, the active current level, and the transient current peaks that are produced by the falling and rising edges of Chip Enable. The magnitude of these transient current peaks is dependent on the output capacitive loading of the device. The associated transient voltage peaks can be suppressed by complying with Intel's Two-Line Control, as detailed in Intel's Application Note AP-72, Order Number 8566, and by properly selected decoupling capacitors. It is recommended that a 0.1 μF ceramic capacitor be used on every device between V_{CC} and GND. This should be a high frequency capacitor of low inherent inductance and should be placed as close to the device as possible. In addition, a 4.7 μF bulk electrolytic capacitor should be used between V_{CC} and GND for every eight devices. The bulk capacitor should be located near where the power supply is connected to the array. The purpose of the bulk capacitor is to overcome the voltage droop caused by the inductive effect of PC board-traces.

PROGRAMMING MODES

Caution: Exceeding 13V on pin 1 (V_{PP}) will permanently damage the 2764A.

Initially, and after each erasure, all bits of the 2764A are in the "1" state. Data is introduced by selectively programming "0s" into the desired bit locations. Although only "0s" will be programmed, both "1s" and "0s" can be present in the data word. The only way to change a "0" to a "1" is by ultraviolet light erasure.

The 2764A is in the programming mode when V_{PP} input is at 12.5V and $\overline{CE}$ and $\overline{PGM}$ are both at TTL low. The data to be programmed is applied 8 bits in parallel to the data output pins. The levels required for the address and data inputs are TTL.

inteligent Programming™ Algorithm

The 2764A inteligent Programming Algorithm rapidly programs Intel 2764A EPROMs using an efficient and reliable method particularly suited to the production programming environment. Typical programming time for individual devices is on the order of one and a half minutes. Programming reliability is also ensured as the incremental program margin of each byte is continually monitored to determine when it has been successfully programmed. A flowchart of the 2764A inteligent Programming Algorithm is shown in Figure 3.

The inteligent Programming Algorithm utilizes two different pulse types: initial and overprogram. The duration of the initial $\overline{CE}$ pulse(s) is one millisecond, which will then be followed by a longer overprogram pulse of length 3X msec. X is an iteration counter and is equal to the number of the initial one millisecond pulses applied to a particular 2764A location, before a correct verify occurs. Up to 25 one-millisecond pulses per byte are provided for before the overprogram pulse is applied.

The entire sequence of program pulses and byte verifications is performed at V_{CC} = 6.0V and V_{PP} = 12.5V. When the inteligent Programming cycle has been completed, all bytes should be compared to the original data with $V_{CC} = V_{PP} = 5.0V$.

Program Inhibit

Programming of multiple 2764As in parallel with different data is easily accomplished by using the Program Inhibit mode. A high-level $\overline{CE}$ or $\overline{PGM}$ input inhibits the other 2764As from being programmed.

Except for $\overline{CE}$, all like inputs (including $\overline{OE}$) of the parallel 2764As may be common. A TTL low-level pulse applied to the $\overline{CE}$ input with V_{PP} at 12.5V will program the selected 2764A.

Verify

A verify should be performed on the programmed bits to determine that they have been correctly programmed. The verify is performed with $\overline{OE}$ at V_{IL}, $\overline{CE}$ at V_{IL}, $\overline{PGM}$ at V_{IH} and V_{PP} at 12.5V.

inteligent Identifier™ Mode

The inteligent Identifier Mode allows the reading out of a binary code from an EPROM that will identify its manufacturer and type. This mode is intended for use by programming equipment for the purpose of automatically matching the device to be programmed with its corresponding programming algorithm. This mode is functional in the 25°C ±5°C ambient temperature range that is required when programming the 2764A.

To activate this mode, the programming equipment must force 11.5V to 12.5V on address line A9 (pin 24) of the 2764A. Two identifier bytes may then be sequenced from the device outputs by toggling address line A0 (pin 10) from V_{IL} to V_{IH}. All other address lines must be held at V_{IL} during inteligent Identifier Mode.

Byte 0 (A0 = V_{IL}) represents the manufacturer code and byte 1 (A0 = V_{IH}) the device identifier code. For the Intel 2764A, these two identifier bytes are given in Table 2. All identifiers for manufacturer and device codes will possess odd parity, with the MSB (O_7) defined as the parity bit.

Table 2. 2764A inteligent Identifier™ Bytes

Identifier \ Pins	A_0 (10)	O_7 (19)	O_6 (18)	O_5 (17)	O_4 (16)	O_3 (15)	O_2 (13)	O_1 (12)	O_0 (11)	Hex Data
Manufacturer Code	V_{IL}	1	0	0	0	1	0	0	1	89
Device Code	V_{IH}	0	0	0	0	1	0	0	0	08

NOTES:
1. $A_9 = 12.0V \pm 0.5V$
2. A_1–A_6, A_{10}–A_{13}, $\overline{CE}$, $\overline{OE} = V_{IL}$
3. $A_{14} = V_{IH}$ or V_{IL}

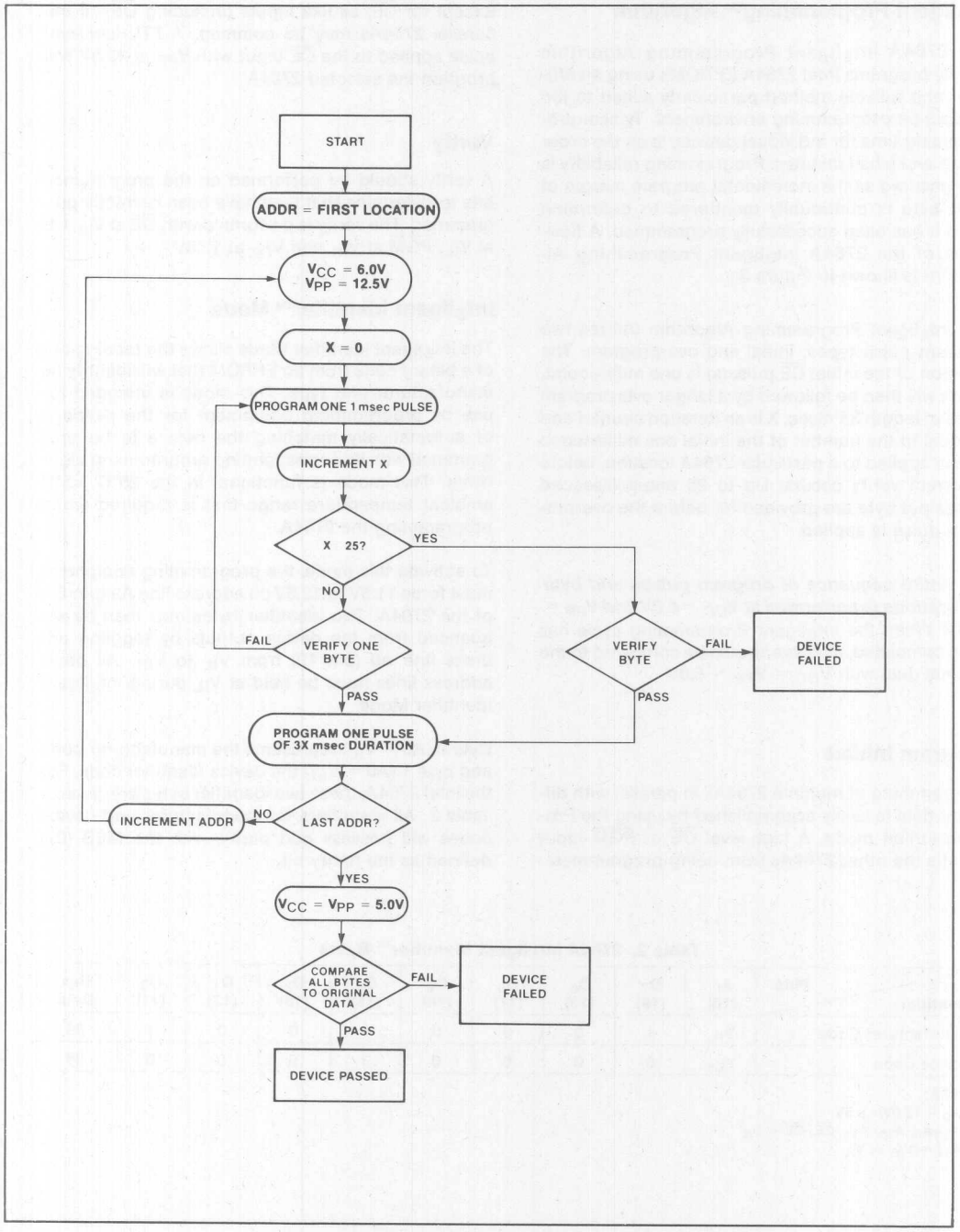


Figure 3. 2764A int$_e$ligent Programming™ Flowchart

ERASURE CHARACTERISTICS

The erasure characteristics of the 2764A are such that erasure begins to occur upon exposure to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000–4000Å range. Data show that constant exposure to room level fluorescent lighting could erase that typical 2764A in approximately 3 years, while it would take approximately 1 week to cause erasure when exposed to direct sunlight. If the 2764A is to be exposed to these types of lighting conditions for extended periods of time, opaque labels should be placed over the 2764A window to prevent unintentional erasure.

The recommended erasure procedure for the 2764A is exposure to shortwave ultraviolet light which has a wavelength of 2537 Angstroms (Å). The integrated dose (i.e., UV intensity × exposure time) for erasure should be a minimum of 15 Wsec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with a 12000 μW/cm^2 power rating. The 2764A should be placed within 1 inch of the lamp tubes during erasure. The maximum integrated dose a 2764A can be exposed to without damage is 7258 Wsec/cm^2 (1 week @ 12000 μW/cm^2). Exposure of the 2764A to high intensity UV light for long periods may cause permanent damage.

RELEVANT INTEL LITERATURE

AR-265 Versatile Algorithm, Equipment Cut Programming Time
RR-35A EPROM Reliability Data Summary

inteligent Programming™ Algorithm

D.C. PROGRAMMING CHARACTERISTICS:
$T_A = 25 \pm 5°C$, $V_{CC} = 6.0V \pm 0.25V$, $V_{PP} = 12.5V \pm 0.3V$

Symbol	Parameter	Limits			Test Conditions (see Note 1)
		Min.	Max.	Unit	
I_{LI}	Input Current (All Inputs)		10	μA	$V_{IN} = V_{IL}$ or V_{IH}
V_{IL}	Input Low Level (All Inputs)	−0.1	0.8	V	
V_{IH}	Input High Level	2.0	$V_{CC}+1$	V	
V_{OL}	Output Low Voltage During Verify		0.45	V	$I_{OL}= 2.1$ mA
V_{OH}	Output High Voltage During Verify	2.4		V	$I_{OH}= -400\ \mu$A
I_{CC2}	V_{CC} Supply Current (Program & Verify)		75	mA	
I_{PP2}	V_{PP} Supply Current (Program)		50	mA	$\overline{CE} = V_{IL}$
V_{ID}	A_9 inteligent Identifier Voltage	11.5	12.5	V	

NOTES:
1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.

A.C. PROGRAMMING CHARACTERISTICS:

$T_A = 25 \pm 5°C$, $V_{CC} = 6.0V \pm 0.25V$, $V_{PP} = 12.5V \pm 0.3V$

Symbol	Parameter	Limits				Test Conditions* (see Note 1)
		Min.	Typ.	Max.	Unit	
t_{AS}	Address Setup Time	2			μs	
t_{OES}	$\overline{OE}$ Setup Time	2			μs	
t_{DS}	Data Setup Time	2			μs	
t_{AH}	Address Hold Time	0			μs	
t_{DH}	Data Hold Time	2			μs	
t_{DFP}[4]	Output Enable to Output Float Delay	0		130	ns	
t_{VPS}	V_{PP} Setup Time	2			μs	
t_{VCS}	V_{CC} Setup Time	2			μs	
t_{PW}	$\overline{PGM}$ Initial Program Pulse Width	0.95	1.0	1.05	ms	(see Note 3)
t_{OPW}	$\overline{PGM}$ Overprogram Pulse Width	2.85		78.75	ms	(see Note 2)
t_{OE}	Data Valid from $\overline{OE}$			150	ns	

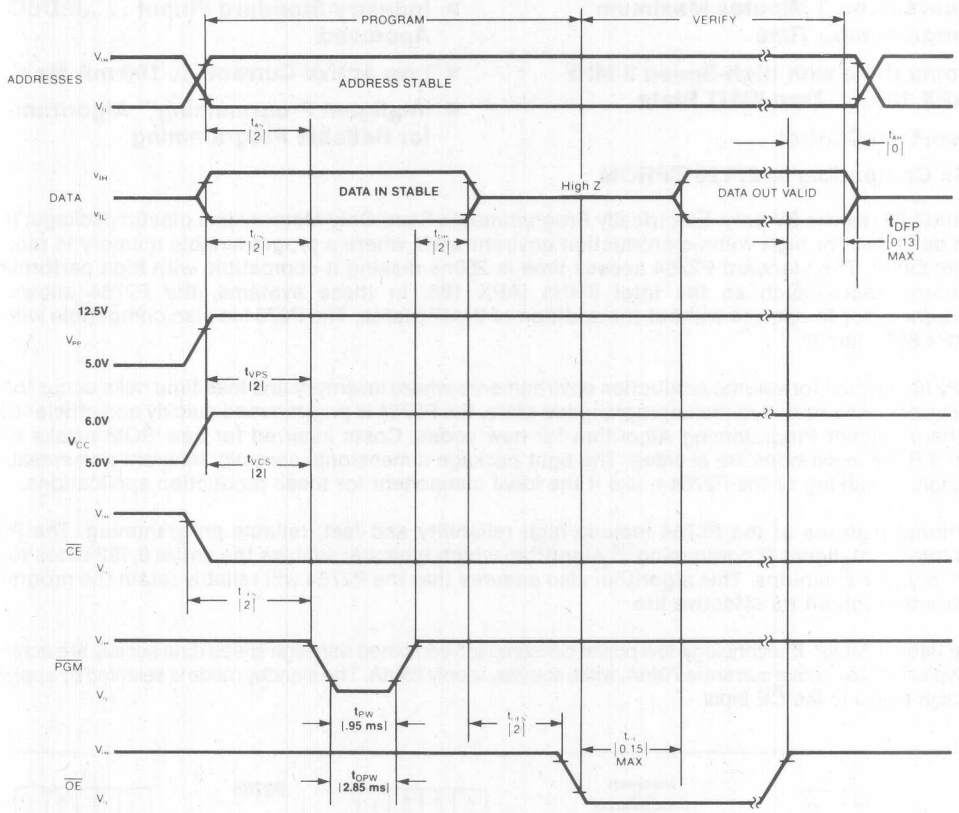
*A.C. CONDITIONS OF TEST

Input Rise and Fall Times (10% to 90%) . . . 20 ns
Input Pulse Levels . . . . . 0.45V to 2.4V
Input Timing Reference Level . . . . . 0.8V and 2.0V
Output Timing Reference Level . . . 0.8V and 2.0V

NOTES:

1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. The length of the overprogram pulse may vary from 2.85 msec to 78.75 msec as a function of the iteration counter value X.
3. Initial Program Pulse width tolerance is 1 msec ±5%.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven—see timing diagram on the following page.

inteligent Programming™ WAVEFORMS



1. ALL TIMES SHOWN IN [ ] ARE MINIMUM AND IN μSEC UNLESS OTHERWISE SPECIFIED.
2. THE INPUT TIMING REFERENCE LEVEL IS .8V FOR V_{IL} AND 2V FOR A V_{IH}.
3. t_{OE} AND t_{DFP} ARE CHARACTERISTICS OF THE DEVICE BUT MUST BE ACCOMMODATED BY THE PROGRAMMER.
4. WHEN PROGRAMMING THE 2764A, A 0.1 μF CAPACITOR IS REQUIRED ACROSS V_{PP} AND GROUND TO SUPRESS SPURIOUS VOLTAGE TRANSIENTS WHICH CAN DAMAGE THE DEVICE.

P2764
64K (8K x 8) PRODUCTION EPROM

- **Guaranteed 2 Minutes Maximum Programming Time**
- **Compatible with High-Speed 8 MHz iAPX 186 . . . Zero WAIT State**
- **Two-Line Control**
- **Pin Compatible to 27128 EPROM**
- **Industry-Standard Pinout . . . JEDEC Approved**
- **Low Active Current . . . 100 mA Max.**
- **int$_e$ligent Programming™ Algorithm—for Reliable Programming**

The Intel P2764 is a 5V-only, Electrically Programmable Read-Only Memory in a plastic package. It has been designed for high volume production environments where a programmable memory is required for flexibility. The standard P2764 access time is 250ns making it compatible with high performance microprocessors, such as the Intel 8MHz iAPX 186. In these systems, the P2764 allows the microprocessor to operate without the addition of WAIT states. The P2764 is also compatible with the 12 MHz 8051 family.

The P2764 is ideal for volume production environments where inventory and lead time risks occur for program codes. Inventoried in the unprogrammed state, the P2764 is programmed quickly and efficiently using the int$_e$ligent Programming Algorithm for new codes. Costs incurred for new ROM masks or obsoleted ROM inventories are avoided. The tight package dimensional controls, inherent non-erasability, and legible marking of the P2764 make it the ideal component for these production applications.

Additional features of the P2764 include high reliability and fast, reliable programming. The P2764 uses Intel's int$_e$ligent Programming™ Algorithm which typically enables the entire 8,192 bytes to program in under 2 minutes. This algorithm also ensures that the P2764 will reliably retain the programmed data throughout its effective life.

Using Intel's HMOS*-E technology, low power consumption combined with high speed data access are achieved. The typical P2764 active current is 70mA, while standby is only 25mA. The standby mode is selected by applying a TTL-high signal to the $\overline{CE}$ input.

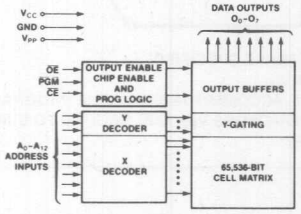
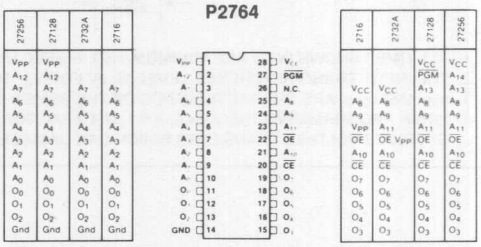


Figure 1. Block Diagram



NOTE: INTEL "UNIVERSAL SITE"-COMPATIBLE EPROM PIN CONFIGURATIONS ARE SHOWN IN THE BLOCKS ADJACENT TO THE P2764 PINS

Figure 2. Pin Configurations

MODE SELECTION

MODE \ PINS	$\overline{CE}$ (20)	$\overline{OE}$ (22)	$\overline{PGM}$ (27)	A_9 (24)	V_{PP} (1)	V_{CC} (28)	Outputs (11-13, 15-19)
Read	V_{IL}	V_{IL}	V_{IH}	X	V_{CC}	V_{CC}	D_{OUT}
Output Disable	V_{IL}	V_{IH}	V_{IH}	X	V_{CC}	V_{CC}	High Z
Standby	V_{IH}	X	X	X	V_{CC}	V_{CC}	High Z
int$_e$ligent Programming	V_{IL}	V_{IH}	V_{IL}	X	V_{PP}	V_{CC}	D_{IN}
Verify	V_{IL}	V_{IL}	V_{IH}	X	V_{PP}	V_{CC}	D_{OUT}
Program Inhibit	V_{IH}	X	X	X	V_{PP}	V_{CC}	High Z
int$_e$ligent Identifier	V_{IL}	V_{IL}	V_{IH}	V_H	V_{CC}	V_{CC}	Code

1. X can be V_{IH} or V_{IL}
2. V_H = 12.0V ± 0.5V

PIN NAMES

A_0-A_{12}	ADDRESSES
$\overline{CE}$	CHIP ENABLE
$\overline{OE}$	OUTPUT ENABLE
O_0-O_7	OUTPUTS
$\overline{PGM}$	PROGRAM
N.C.	NO CONNECT

*HMOS is a patented process of Intel Corporation

APRIL 1983
ORDER NUMBER: 230629-001

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias ..... −10°C to +80°C
Storage Temperature ..... −65°C to +125°C
All Input or Output Voltages with Respect to Ground ..... +7.0V to −0.6V
Voltage on Pin 24 with Respect to Ground ..... +13.5V to −0.6V
V_{PP} Supply Voltage with Respect to Ground During Programming ..... +22V to −0.6V

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. AND A.C. OPERATING CONDITIONS DURING READ

	P2764	P2764-3	P2764-4
Operating Temperature Range	0°C–70°C	0°C–70°C	0°C–70°C
V_{CC} Power Supply[1,2]	5V ± 5%	5V ± 5%	5V ± 5%
V_{PP} Voltage[2]	$V_{PP} = V_{CC}$	$V_{PP} = V_{CC}$	$V_{PP} = V_{CC}$

READ OPERATION

D.C. CHARACTERISTICS

Symbol	Parameter	Limits				Conditions
		Min	Typ[3]	Max	Unit	
I_{LI}	Input Load Current			10	μA	$V_{IN} = 5.5V$
I_{LO}	Output Leakage Current			10	μA	$V_{OUT} = 5.5V$
I_{PP1}[2]	V_{PP} Current Read			5	mA	$V_{PP} = 5.5V$
I_{CC1}[2]	V_{CC} Current Standby		25	40	mA	$\overline{CE} = V_{IH}$
I_{CC2}[2]	V_{CC} Current Active		70	100	mA	$\overline{CE} = \overline{OE} = V_{IL}$
V_{IL}	Input Low Voltage	−.1		+.8	V	
V_{IH}	Input High Voltage	2.0		$V_{CC}+1$	V	
V_{OL}	Output Low Voltage			.45	V	$I_{OL} = 2.1$ mA
V_{OH}	Output High Voltage	2.4			V	$I_{OH} = -400\ \mu A$

A.C. CHARACTERISTICS

Symbol	Parameter	P2764 Limits		P2764-3 Limits		P2764-4 Limits		Unit	Test Conditions
		Min	Max	Min	Max	Min	Max		
t_{ACC}	Address to Output Delay		250		300		450	ns	$\overline{CE}=\overline{OE}=V_{IL}$
t_{CE}	$\overline{CE}$ to Output Delay		250		300		450	ns	$\overline{OE}=V_{IL}$
t_{OE}	$\overline{OE}$ to Output Delay		100		120		150	ns	$\overline{CE}=V_{IL}$
t_{DF}[4]	$\overline{OE}$ High to Output Float	0	60	0	105	0	130	ns	$\overline{CE}=V_{IL}$
t_{OH}	Output Hold from Addresses, $\overline{CE}$ or $\overline{OE}$ Whichever Occurred First	0		0		0		ns	$\overline{CE}=\overline{OE}=V_{IL}$

NOTES:

1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. V_{PP} may be connected directly to V_{CC} except during programming. The supply current would then be the sum of I_{CC} and I_{PP1}.
3. Typical values are for $t_A = 25°C$ and nominal supply voltages.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven—see timing diagram on the following page.

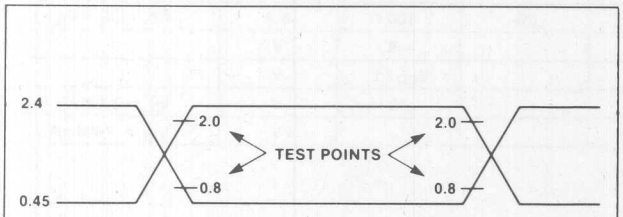
CAPACITANCE ($T_A = 25°C$, f = 1MHz)

Symbol	Parameter	Typ.[1]	Max.	Unit	Conditions
C_{IN}[2]	Input Capacitance	4	6	pF	$V_{IN}=0V$
C_{OUT}	Output Capacitance	8	12	pF	$V_{OUT}=0V$

NOTES:

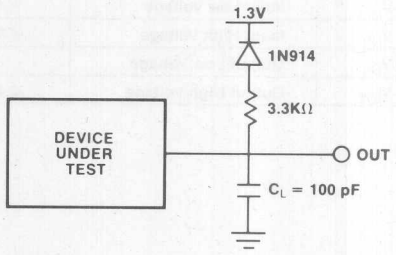
1. Typical values are for $T_A = 25°C$ and nominal supply voltages.
2. This parameter is only sampled and is not 100% tested.

A.C. TESTING INPUT/OUTPUT WAVEFORM



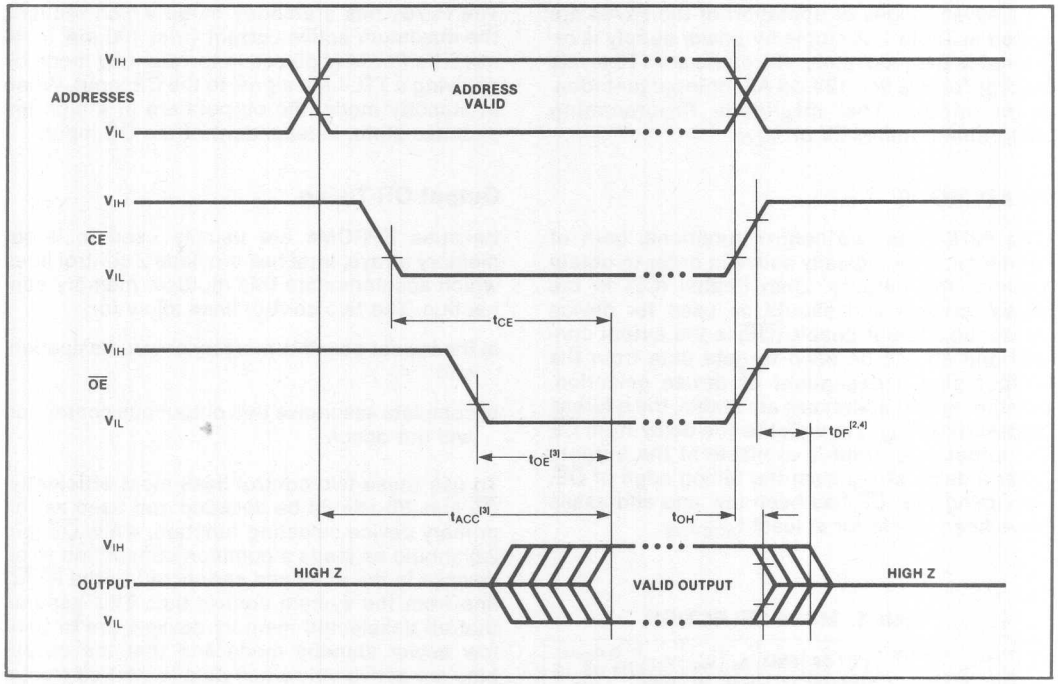
A.C. TESTING: INPUTS ARE DRIVEN AT 2.4V FOR A LOGIC "1" AND 0.45V FOR A LOGIC "0." TIMING MEASUREMENTS ARE MADE AT 2.0V FOR A LOGIC "1" AND 0.8V FOR A LOGIC "0".

A.C. TESTING LOAD CIRCUIT



C_L = 100 pF
C_L INCLUDES JIG CAPACITANCE

A.C. WAVEFORMS



NOTES:

1. Typical values are for T_A = 25°C and nominal supply voltages.
2. This parameter is only sampled and is not 100% tested.
3. OE may be delayed up to $t_{ACC} - t_{OE}$ after the falling edge of CE without impact on t_{ACC}.
4. t_{DF} is specified from OE or CE, whichever occurs first.

APPLICATIONS INFORMATION

Intel's P2764 is the result of a multi-year effort to make EPROMs more cost effective for production applications. The benefits of a plastic package enable the P2764 to be used for high volume production with lower profile boards and easier production assembly (no cover over UV transparent windows).

The reliability of plastic EPROMs is equivalent to traditional CERDIP packaging. The plastic is rugged and durable making it optimal for auto insertion and auto handling equipment. Design and testing ensures device programmability, data integrity, and impermeability to moisture.

Intel's Plastic EPROMs are designed for total compatibility with their CERDIP packaged predecessors. This encompasses quality, reliability, and programming. All Intel Plastic EPROMs have passed Intel's strict process and product reliability qualifications.

DEVICE OPERATION

The seven modes of operation of the P2764 are listed in Table 1. A single 5V power supply is required in the read mode. All inputs are TTL levels except for V_{PP} and 12V on A9 for int$_e$ligent Identifier mode. The int$_e$ligent Programming Algorithm requires 6V on V_{CC}.

READ MODE

The P2764 has two control functions, both of which must be logically active in order to obtain data at the outputs. Chip Enable ($\overline{CE}$) is the power control and should be used for device selection. Output Enable ($\overline{OE}$) is the output control and should be used to gate data from the output pins, independent of device selection. Assuming that addresses are stable, the address access time (t_{ACC}) is equal to the delay from $\overline{CE}$ to output (t_{CE}). Data is available at the outputs after a delay of t_{OE} from the falling edge of $\overline{OE}$, assuming that $\overline{CE}$ has been low and addresses have been stable for at least t_{ACC}-t_{OE}.

Table 1. MODE SELECTION

MODE \ PINS	$\overline{CE}$ (20)	$\overline{OE}$ (22)	$\overline{PGM}$ (27)	A_9 (24)	V_{PP} (1)	V_{CC} (28)	Outputs (11–13, 15–19)
Read	V_{IL}	V_{IL}	V_{IH}	X	V_{CC}	V_{CC}	D_{OUT}
Output Disable	V_{IL}	V_{IH}	V_{IH}	X	V_{CC}	V_{CC}	High Z
Standby	V_{IH}	X	X	X	V_{CC}	V_{CC}	High Z
Verify	V_{IL}	V_{IL}	V_{IH}	X	V_{PP}	V_{CC}	D_{OUT}
Program Inhibit	V_{IH}	X	X	X	V_{PP}	V_{CC}	High Z
int$_e$ligent Identifier	V_{IL}	V_{IL}	V_{IH}	V_H	V_{CC}	V_{CC}	Code
int$_e$ligent Programming	V_{IL}	V_{IH}	V_{IL}	X	V_{PP}	V_{CC}	D_{IN}

NOTES:
1. X can be V_{IH} or V_{IL}
2. V_H = 12.0V ±0.5V

STANDBY MODE

The P2764 has a standby mode which reduces the maximum active current from 100 mA to 40 mA. The P2764 is placed in the standby mode by applying a TTL-high signal to the $\overline{CE}$ input. When in standby mode, the outputs are in a high impedance state, independent of the $\overline{OE}$ input.

Output OR-Tieing

Because EPROMs are usually used in larger memory arrays, Intel has provided 2 control lines which accommodate this multiple memory connection. The two control lines allow for:

a) the lowest possible memory power dissipation, and

b) complete assurance that output bus contention will not occur.

To use these two control lines most efficiently, $\overline{CE}$ (pin 20) should be decoded and used as the primary device selecting function, while $\overline{OE}$ (pin 22) should be made a common connection to all devices in the array and connected to the $\overline{READ}$ line from the system control bus. This assures that all deselected memory devices are in their low power standby mode and that the output pins are active only when data is desired from a particular memory device.

System Considerations

The power-switching characteristics of HMOS-E EPROMs require careful decoupling of the devices. The supply current, I_{CC}, has three segments that are of interest to the system designer — the standby current level, the active current level, and the transient current peaks that are produced by the falling and rising edges of Chip Enable. The magnitude of these tran-

sient current peaks is dependent on the output capacitive loading of the device. The associated transient voltage peaks can be suppressed by complying with Intel's Two-Line Control, as detailed in Intel's Application Note, AP-72, and by properly selected decoupling capacitors. It is recommended that a 0.1μF cermic capacitor be used on every device between V_{CC} and GND. This should be a high-frequency capacitor of low inherent inductance and should be placed as close to the device as possible. In addition, a 4.7μF bulk electrolytic capacitor should be used between V_{CC} and GND for every eight devices. The bulk capacitor should be located near where the power supply is connected to the array. The purpose of the bulk capacitor is to overcome the voltage droop caused by the inductive effect of PC board-traces.

PROGRAMMING MODES

CAUTION: The P2764 must be programmed using the int$_e$ligent Programming Algorithm. Exceeding 22V on pin 1 (V_{PP}) will permanently damage the P2764.

Initially, all bits of the P2764 are in the "1" state. Data is introduced by selectively programming "0s" into the desired bit locations. Although only "0s" will be programmed, both "1s" and "0s" can be present in the data word.

The P2764 is in the programming mode when the V_{PP} Input is at 21V and $\overline{CE}$ and $\overline{PGM}$ are both at TTL-low. The data to be programmed is applied 8 bits in parallel to the data output pins. The levels required for the address and data inputs are TTL.

int$_e$ligent Programming™ Algorithm

The int$_e$ligent Programming Algorithm allows the P2764 to be programmed in a significantly faster time than standard 2764 EPROMs. Typical int$_e$ligent programming times are on the order of a minute and a half. This is a five-fold reduction in programming time from a standard 50 msec method. A flowchart of the int$_e$ligent Programming Algorithm is shown in Fig. 3.

Execution of the int$_e$ligent Programming Algorithm at maximum speed will result in a guaranteed maximum programming time of two minutes for the P2764 when programmed individually. This excludes the time for initial erase check, final verify, and programmer overhead.

This fast algorithm results in high reliability characteristics through the "closed loop" technique of margin checking. The P2764 is designed to retain data for at least 20 years.

To ensure reliable program margin, the int$_e$ligent Programming Algorithm utilizes two different pulse types: initial and overprogram. The duration of the initial $\overline{PGM}$ pulse (s) is one millisecond, which will then be followed by a longer overprogram pulse of length 4X msec. X is an iteration counter and is equal to the number of the initial one millisecond pulses applied to a particular P2764 location, before a correct verify occurs. Up to 15 one-millisecond pulses per byte are provided for before the overprogram pulse is applied.

The entire sequence of program pulses and byte verifications is performed at V_{CC} = 6.0V and V_{PP} = 21.0V. When the int$_e$ligent Programming cycle has been completed, all bytes should be compared to the original data with $V_{CC} = V_{PP} = 5.0V$.

Program Inhibit

Programming of multiple P2764s in parallel with different data is also easily accomplished by using the Program inhibit mode. A high-level $\overline{CE}$ or $\overline{PGM}$ input inhibits the other P2764s from being programmed. Except for $\overline{CE}$ (or $\overline{PGM}$), all like inputs (including $\overline{OE}$) of the parallel F2764s may be common. A TTL low-level pulse applied to a P2764 $\overline{CE}$ and $\overline{PGM}$ input with V_{PP} at 21V will program that P2764.

Program Verify

A verify should be performed on the programmed bits to determine that they were correctly programmed. The verify is accomplished with $\overline{CE}$ and $\overline{OE}$ at V_{IL}. However, $\overline{PGM}$ is at V_{IH}.

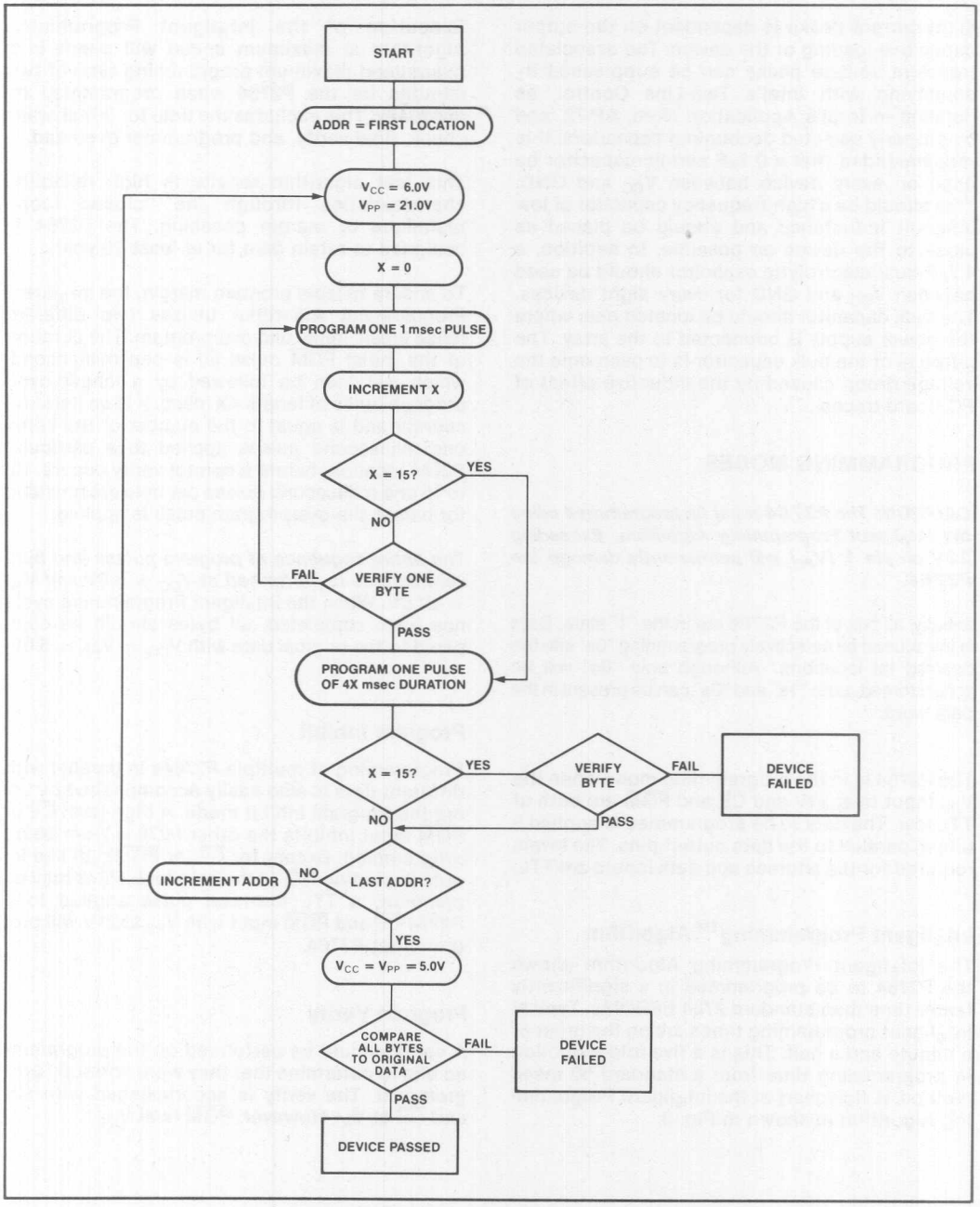


Figure 3. P2764 int$_e$ligent Programming™ Flowchart

inteligent Programming™ Algorithm

D.C. PROGRAMMING CHARACTERISTICS: $T_A = 25 \pm 5°C$, $V_{CC} = 6.0V \pm 0.25V$, $V_{PP} = 21V \pm 0.5V$ (see Note 1)

Symbol	Parameter	Limits			Test Conditions
		Min.	Max.	Unit	
I_{LI}	Input Current (All Inputs)		10	μA	$V_{IN} = V_{IL}$ or V_{IH}
V_{IL}	Input Low Level (All Inputs)	−0.1	0.8	V	
V_{IH}	Input High Level	2.0	$V_{CC}+1$	V	
V_{OL}	Output Low Voltage During Verify		0.45	V	I_{OL} = 2.1 mA
V_{OH}	Output High Voltage During Verify	2.4		V	$I_{OH} = -400\ \mu A$
I_{CC2}	V_{CC} Supply Current (Program & Verify)		100	mA	
I_{PP2}	V_{PP} Supply Current (Program)		30	mA	$\overline{CE} = V_{IL} = \overline{PGM}$
V_{ID}	A9 inteligent Identifier Voltage	11.5	12.5	V	

A.C. PROGRAMMING CHARACTERISTICS: $T_A = 25 \pm 5°C$, $V_{CC} = 6.0V \pm 0.25V$, $V_{PP} = 21V \pm 0.5V$ (see Note 1)

Symbol	Parameter	Limits				Test Conditions*
		Min.	Typ.	Max.	Unit	
t_{AS}	Address Setup Time	2			μs	
t_{OES}	$\overline{OE}$ Setup Time	2			μs	
t_{DS}	Data Setup Time	2			μs	
t_{AH}	Address Hold Time	0			μs	
t_{DH}	Data Hold Time	2			μs	
t_{DFP}	Output Enable to Output Float Delay	0		130	ns	
t_{VPS}	V_{PP} Setup Time	2			μs	
t_{VCS}	V_{CC} Setup Time	2			μs	
t_{PW}	$\overline{PGM}$ Initial Program Pulse Width	0.95	1.0	1.05	ms	(see Note 3)
t_{OPW}	$\overline{PGM}$ Overprogram Pulse Width	3.8		63	ms	(see Note 2)
t_{CES}	$\overline{CE}$ Setup Time	2			μs	
t_{OE}	Data Valid from $\overline{OE}$			150	ns	

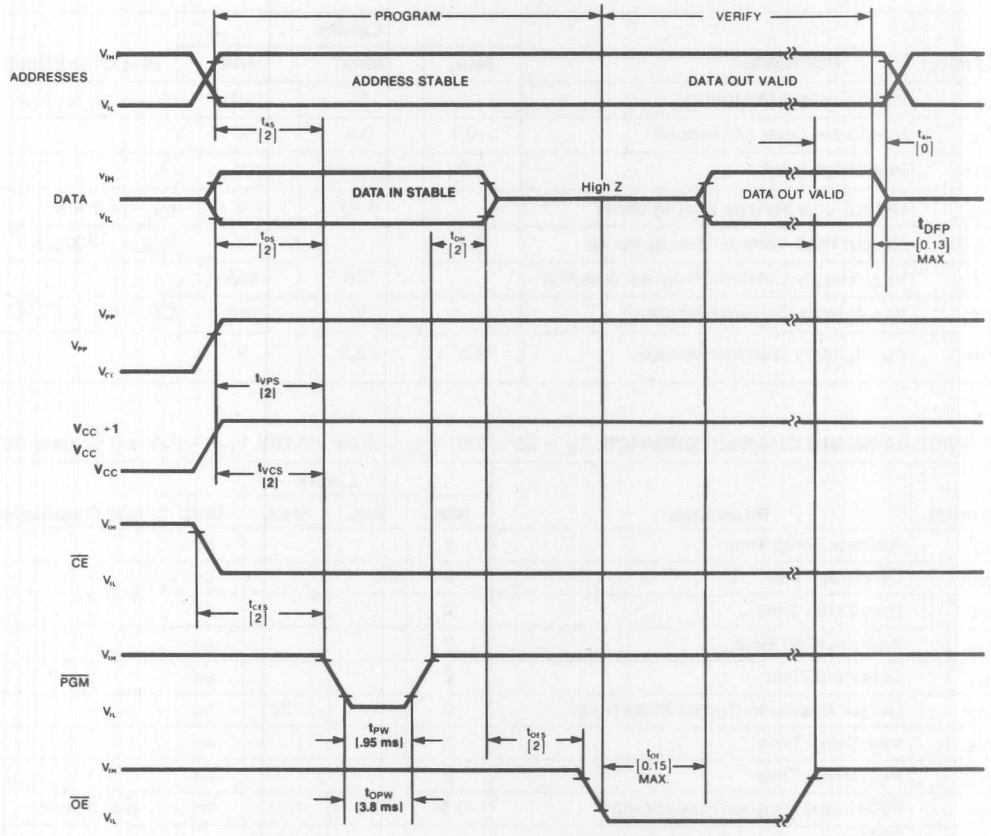
***A.C. CONDITIONS OF TEST**

Input Rise and Fall Times (10% to 90%) ..... 20 ns
Input Pulse Levels ..... 0.45V to 2.4V
Input Timing Reference Level ..... 0.8V and 2.0V
Output Timing Reference Level ..... 0.8V and 2.0V

NOTES:

1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. The length of the overprogram pulse will vary from 3.8 msec to 63 msec as a function of the iteration counter value X.
3. Initial Program Pulse width tolerance is 1 msec ± 5%.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven—see timing diagram on the following page.

int$_e$ligent Programming™ WAVEFORMS



1. ALL TIMES SHOWN IN [ ] ARE MINIMUM AND IN μSEC UNLESS OTHERWISE SPECIFIED.
2. THE INPUT TIMING REFERENCE LEVEL IS .8V FOR V_{IL} AND 2V FOR A V_{IH}.
3. t_{OE} AND t_{DFP} ARE CHARACTERISTICS OF THE DEVICE BUT MUST BE ACCOMMODATED BY THE PROGRAMMER.
4. WHEN PROGRAMMING THE P2764, A 0.1μ CAPACITOR IS REQUIRED ACROSS V_{PP} AND GROUND TO SUPPRESS SPURIOUS VOLTAGE TRANSIENTS WHICH CAN DAMAGE THE DEVICE.

inteligent Identifier™ Mode

The inteligent Identifier Mode allows the reading out of a binary code from an EPROM that will identify its manufacturer and type. This mode is intended for use by programming equipment for the purpose of automatically matching the device to be programmed with its corresponding programming algorithm. This mode is functional in the 25°C ± 5°C ambient temperature range.

To activate this mode, the programming equipment must force 11.5V to 12.5V on address line A9 (pin 24) of the P2764. Two identifier bytes may then be sequenced from the device outputs by toggling address line A0 (pin 10) from V_{IL} to V_{IH}. All other address lines must be held at V_{IL} during the inteligent Identifier mode.

Byte 0 (A0 = V_{IL}) represents the manufacturer code and byte 1 (A0 = V_{IH}) the device identifier code. For the Intel P2764, these two identifier bytes are given in Table 2. These are also the same identifier bytes as the Intel 2764 EPROM. All identifiers for manufacturer and device codes will possess odd parity, with the MSB (O_7) defined as the parity bit.

Note: In programming equipment that employs the inteligent Identifier function, a provision must be made to ensure that the identifier bytes select only the inteligent Programming™ Algorithm to program the P2764, or that the function can be manually bypassed to do so.

Table 2. P2764 inteligent Identifer™ Bytes

Identifier \ Pins	A_0 (10)	O_7 (19)	O_6 (18)	O_5 (17)	O_4 (16)	O_3 (15)	O_2 (13)	O_1 (12)	O_0 (11)	Hex Data
Manufacturer Code	V_{IL}	1	0	0	0	1	0	0	1	89
Device Code	V_{IH}	0	0	0	0	0	0	1	0	02

PRELIMINARY

27256
256K (32K x 8) UV ERASABLE PROM

- **Software Carrier Capability**
- **250 ns Maximum Access Time**
- **Two-Line Control**
- **int$_e$ligent Identifier™ Mode**
  —Automated Programming Operations
- **TTL Compatible**
- **Industry Standard Pinout . . . JEDEC Approved**
- **Low Power**
  —100 mA max. Active
  — 40 mA max. Standby
- **int$_e$ligent Programming™ Algorithm**
  —Fastest EPROM Programming

The Intel 27256 is a 5V only, 262,144-bit ultraviolet Erasable and Electrically Programmable Read Only Memory (EPROM). Organized as 32K words by 8 bits, individual bytes are accessed in under 250ns. This is compatible with high performance microprocessors, such as the Intel 8MHz iAPX 186, allowing full speed operation without the addition of performance-degrading WAIT states. The 27256 is also directly compatible with Intel's 8051 family of microcontrollers.

The 27256 enables implementation of new, advanced systems with firmware intensive architectures. The combination of the 27256's high density, cost effective EPROM storage, and new advanced microprocessors having megabit addressing capability provides designers with opportunities to engineer user-friendly, high reliability, high-performance systems.

The 27256's large storage capability of 32K bytes enables it to function as a high density software carrier. Entire operating systems, diagnostics, high-level language programs and specialized application software can reside in a 27256 EPROM directly on a system's memory bus. This permits immediate microprocessor access and execution of software and eliminates the need for time consuming disk accesses and downloads.

Several advanced features have been designed into the 27256 that allow for fast and reliable programming —the int$_e$ligent identifier™ mode and the int$_e$ligent Programming™ Algorithm. Programming equipment that takes advantage of these innovations will electronically identify the 27256 and then rapidly program it using an efficient programming method.

Two-line control and JEDEC-approved, 28-pin packaging are standard features of all Intel high-density EPROMs. This assures easy microprocessor interfacing and minimum design efforts when upgrading, adding, or choosing between nonvolatile memory alternatives.

The 27256 is manufactured using Intel's advanced HMOS*II-E technology.

*HMOS is a patented process of Intel Corporation.

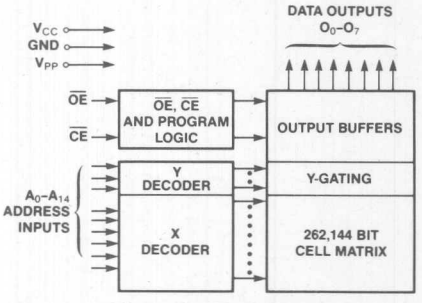
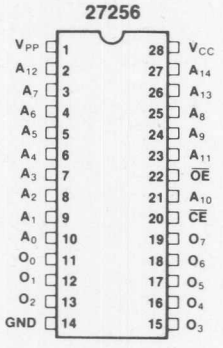


Figure 1. Block Diagram



PIN NAMES	
A_0–A_{14}	ADDRESSES
$\overline{CE}$	CHIP ENABLE
$\overline{OE}$	OUTPUT ENABLE
O_0–O_7	OUTPUTS

Figure 2. Pin Configuration

JUNE 1983

ORDER NUMBER: 210827-005

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias ..... −10°C to +80°C
Storage Temperature ..... −65°C to +125°C
All Input or Output Voltages with
Respect to Ground .....+6.25 V to −0.6V
Voltage on Pin 24 with
Respect to Ground .....+13.5V to −0.6V
V_{PP} Supply Voltage with Respect
to Ground .....+13.0 V to −0.6V

**NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

D.C. AND A.C. OPERATING CONDITIONS DURING READ

	27256	27256-3	27256-4	27256-25	27256-30	27256-45
Operating Temperature Range	0°C–70°C	0°C–70°C	0°C–70°C	0°C–70°C	0°C–70°C	0°C–70°C
V_{CC} Power Supply[1,2]	5V ± 5%	5V ± 5%	5V ± 5%	5V ± 10%	5V ± 10%	5V ± 10%

READ OPERATION

A.C. CHARACTERISTICS

Symbol	Parameter	Limits			Units	Test Conditions
		Min.	Typ.[3]	Max.		
I_{LI}	Input Load Current			10	µA	V_{IN} = 5.5V
I_{LO}	Output Leakage Current			10	µA	V_{OUT} = 5.5V
I_{PP1}[2]	V_{PP} Current Read/Standby			5	mA	V_{PP} = 5.5V
I_{CC1}[2]	V_{CC} Current Standby		20	40	mA	$\overline{CE}$ = V_{IH}
I_{CC2}[2]	V_{CC} Current Active		45	100	mA	$\overline{CE}$ = $\overline{OE}$ = V_{IL} V_{PP} = V_{CC}
V_{IL}	Input Low Voltage	−.1		+.8	V	
V_{IH}	Input High Voltage	2.0		V_{CC}+1	V	
V_{OL}	Output Low Voltage			.45	V	I_{OL} = 2.1 mA
V_{OH}	Output High Voltage	2.4			V	I_{OH} = −400 µA
V_{PP}[2]	V_{PP} Read Voltage	3.8		V_{CC}	V	V_{CC} = 5.0V ± 0.25V

READ OPERATION

A.C. CHARACTERISTICS

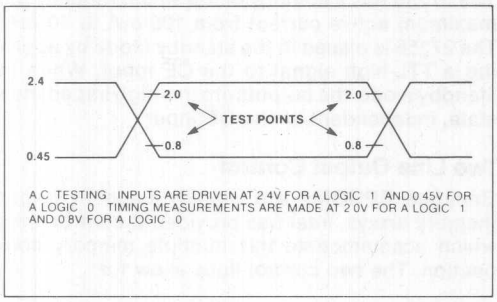
Symbol	Parameter	27256-25 & 27256 Limits		27256-30 & 27256-3 Limits		27256-45 & 27256-4 Limits		Units	Test Conditions
		Min.	Max.	Min.	Max.	Min.	Max.		
t_{ACC}	Address to Output Delay		250		300		450	ns	$\overline{CE}=\overline{OE}=V_{IL}$
t_{CE}	$\overline{CE}$ to Output Delay		250		300		450	ns	$\overline{OE}=V_{IL}$
t_{OE}	$\overline{OE}$ to Output Delay		100		120		150	ns	$\overline{CE}=V_{IL}$
t_{DF}[4]	$\overline{OE}$ High to Output Float	0	60	0	105	0	130	ns	$\overline{CE}=V_{IL}$
t_{OH}	Output Hold from Addresses, $\overline{CE}$ or $\overline{OE}$ Whichever Occurred First	0		0		0		ns	$\overline{CE}=\overline{OE}=V_{IL}$

NOTES:
1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. V_{PP} may be connected directly to V_{CC} except during programming. The supply current would then be the sum of I_{CC} and I_{PP1}.
3. Typical values are for t_A =25°C and nominal supply voltages.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven—see timing diagram

CAPACITANCE (T_A = 25°C, f = 1 MHz)

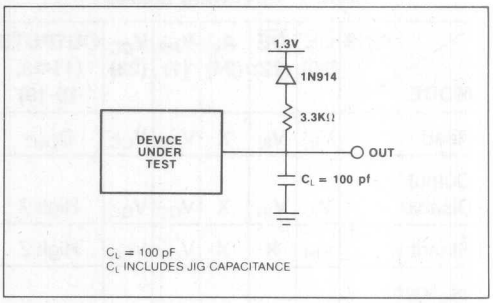
Symbol	Parameter	Typ.[1]	Max.	Unit	Conditions
C_{IN}[2]	Input Capacitance	4	6	pF	V_{IN}=0V
C_{OUT}	Output Capacitance	8	12	pF	V_{OUT}=0V

A.C. TESTING INPUT/OUTPUT WAVEFORM



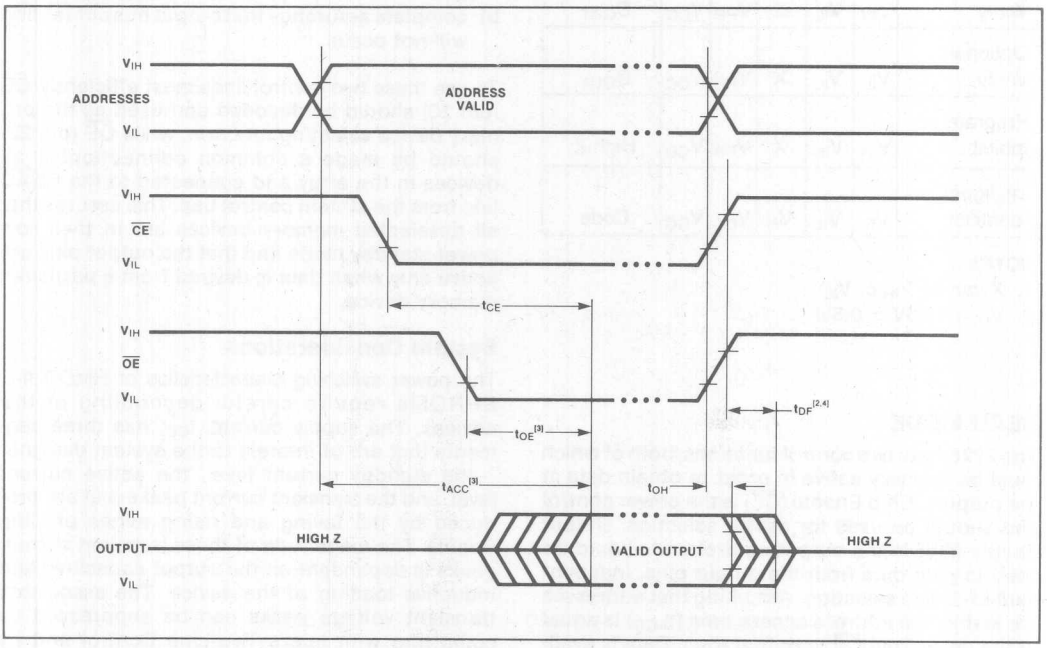
A C TESTING INPUTS ARE DRIVEN AT 2 4V FOR A LOGIC 1 AND 0 45V FOR A LOGIC 0 TIMING MEASUREMENTS ARE MADE AT 2 0V FOR A LOGIC 1 AND 0 8V FOR A LOGIC 0

A.C. TESTING LOAD CIRCUIT



C_L = 100 pF
C_L INCLUDES JIG CAPACITANCE

A.C. WAVEFORMS



NOTES:
1. Typical values are for T_A = 25°C and nominal supply voltages.
2. This parameter is only sampled and is not 100% tested.
3. $\overline{OE}$ may be delayed up to $t_{ACC} - t_{OE}$ after the falling edge of $\overline{CE}$ without impact on t_{ACC}.
4. t_{DF} is specified from $\overline{OE}$ or $\overline{CE}$, whichever occurs first.

DEVICE OPERATION

The eight modes of operation of the 27256 are listed in Table 1. A single 5V power supply is required in the read mode. All inputs are TTL levels except for V_{PP} and 12V on A9 for int$_e$ligent identifier mode.

Table 1. Operating Modes

MODE \ PINS	$\overline{CE}$ (20)	$\overline{OE}$ (22)	A_9 (24)	V_{PP} (1)	V_{CC} (28)	OUTPUTS (11-13, 15-19)
Read	V_{IL}	V_{IL}	X	V_{CC}	V_{CC}	D_{OUT}
Output Disable	V_{IL}	V_{IH}	X	V_{CC}	V_{CC}	High Z
Standby	V_{IH}	X	X	V_{CC}	V_{CC}	High Z
int$_e$ligent Programming	V_{IL}	V_{IH}	X	V_{PP}	V_{CC}	D_{IN}
Verify	V_{IH}	V_{IL}	X	V_{PP}	V_{CC}	D_{OUT}
Optional Verify	V_{IL}	V_{IL}	X	V_{PP}	V_{CC}	D_{OUT}
Program Inhibit	V_{IH}	V_{IH}	X	V_{PP}	V_{CC}	High Z
int$_e$ligent Identifier	V_{IL}	V_{IL}	V_H	V_{CC}	V_{CC}	Code

NOTES:
1. X can be V_{IH} or V_{IL}
2. $V_H = 12.0V \pm 0.5V$

READ MODE

The 27256 has two control functions, both of which must be logically active in order to obtain data at the outputs. Chip Enable ($\overline{CE}$) is the power control and should be used for device selection. Output Enable ($\overline{OE}$) is the output control and should be used to gate data from the output pins, independent of device selection. Assuming that addresses are stable, the address access time (t_{ACC}) is equal to the delay from $\overline{CE}$ to output (t_{CE}). Data is available at the outputs after a delay of t_{OE} from the falling edge of $\overline{OE}$, assuming that $\overline{CE}$ has been low and addresses have been stable for at least $t_{ACC}-t_{OE}$.

STANDBY MODE

The 27256 has a standby mode which reduces the maximum active current from 100 mA to 40 mA. The 27256 is placed in the standby mode by applying a TTL-high signal to the $\overline{CE}$ input. When in standby mode, the outputs are in a high impedance state, independent of the $\overline{OE}$ input.

Two Line Output Control

Because EPROMs are usually used in larger memory arrays, Intel has provided 2 control lines which accommodate this multiple memory connection. The two control lines allow for:

a) the lowest possible memory power dissipation, and
b) complete assurance that output bus contention will not occur.

To use these two control lines most efficiently, $\overline{CE}$ (pin 20) should be decoded and used as the primary device selecting function, while $\overline{OE}$ (pin 22) should be made a common connection to all devices in the array and connected to the $\overline{READ}$ line from the system control bus. This assures that all deselected memory devices are in their low power standby mode and that the output pins are active only when data is desired from a particular memory device.

System Considerations

The power switching characteristics of HMOS II-E EPROMs require careful decoupling of the devices. The supply current, I_{CC}, has three segments that are of interest to the system designer —the standby current level, the active current level, and the transient current peaks that are produced by the falling and rising edges of Chip Enable. The magnitude of these transient current peaks is dependent on the output capacitive and inductive loading of the device. The associated transient voltage peaks can be suppressed by complying with Intel's Two-Line Control and by

properly selected decoupling capacitors. It is recommended that a 0.1 μF ceramic capacitor be used on every device between V_{CC} and GND. This should be a high frequency capacitor of low inherent inductance and should be placed as close to the device as possible. In addition, a 4.7 μF bulk electrolytic capacitor should be used between V_{CC} and GND for every eight devices. The bulk capacitor should be located near where the power supply is connected to the array. The purpose of the bulk capacitor is to overcome the voltage droop caused by the inductive effects of PC board traces.

PROGRAMMING

Caution: Exceeding 13.0V on pin 1 (V_{PP}) will permanently damage the 27256.

Initially, and after each erasure, all bits of the 27256 are in the "1" state. Data is introduced by selectively programming "0s" into the desired bit locations. Although only "0s" will be programmed, both "1s" and "0s" can be present in the data word. The only way to change a "0" to a "1" is by ultraviolet light erasure.

The 27256 is in the programming mode when the V_{PP} input is at 12.5V and $\overline{CE}$ is at TTL-low. The data to be programmed is applied 8 bits in parallel to the data output pins. The levels required for the address and data inputs are TTL.

int$_e$ligent Programming™ Algorithm

The 27256 int$_e$ligent Programming Algorithm rapidly programs Intel 27256 EPROMS using an efficient and reliable method particularly suited to the production programming environment. Typical programming times for individual devices are on the order of five minutes. Programming reliability is also ensured as the incremental program margin of each byte is continually monitored to determine when it has been successfully programmed. A flowchart of the 27256 int$_e$ligent Programming Algorithm is shown in Figure 3.

The int$_e$ligent Programming Algorithm utilizes two different pulse types: initial and overprogram. The duration of the initial $\overline{CE}$ pulse(s) is one millisecond, which will then be followed by a longer overprogram pulse of length 3X msec. X is an iteration counter and is equal to the number of the initial one millisecond pulses applied to a particular 27256 location, before a correct verify occurs. Up to 25 one-millisecond pulses per byte are provided for before the overprogram pulse is applied.

The entire sequence of program pulses and byte verifications is performed at V_{CC} = 6.0 V and V_{PP} = 12.5V. When the int$_e$ligent Programming cycle has been completed, all bytes should be compared to the original data with V_{CC} = V_{PP} = 5.0V.

Program Inhibit

Programming of multiple 27256s in parallel with different data is easily accomplished by using the Program Inhibit mode. A high-level $\overline{CE}$ input inhibits the other 27256s from being programmed.

Except for $\overline{CE}$ and $\overline{OE}$, all like inputs of the parallel 27256s may be common. A TTL low-level pulse applied to the $\overline{CE}$ input with V_{PP} at 12.5V will program the selected 27256.

Verify

A verify should be performed on the programmed bits to determine that they have been correctly programmed. The verify is performed with $\overline{OE}$ at V_{IL}, $\overline{CE}$ at V_{IH} and V_{PP} at 12.5V.

Optional Verify

The optional verify may be performed in place of the verify mode. It is performed with $\overline{OE}$ at V_{IL}, $\overline{CE}$ at V_{IL} (as opposed to the standard verify which has $\overline{CE}$ at V_{IH}), and V_{PP} at 12.5V. The outputs will tri-state according to the signal presented to $\overline{OE}$. Therefore, all devices with V_{PP}=12.5V and $\overline{OE}$=V_{IL} will present data on the bus independent of the $\overline{CE}$ state. When parallel programming several devices which share a common bus, V_{PP} should be lowered to V_{CC} (=6.0V) and the normal read mode used to execute a program verify.

int$_e$ligent Identifier™ Mode

The int$_e$ligent Identifier Mode allows the reading out of a binary code from an EPROM that will identify its manufacturer and type. This mode is intended for use by programming equipment for the purpose of automatically matching the device to be programmed with its corresponding programming algorithm. This mode is functional in the 25°C ± 5°C ambient temperature range that is required when programming the 27256.

To activate this mode, the programming equipment must force 11.5V to 12.5V on address line A9 (pin 24) of the 27256. Two identifier bytes may then be sequenced from the device outputs by toggling address line A0 (pin 10) from V_{IL} to V_{IH}. All other address lines must be held at V_{IL} during int$_e$ligent Identifier Mode.

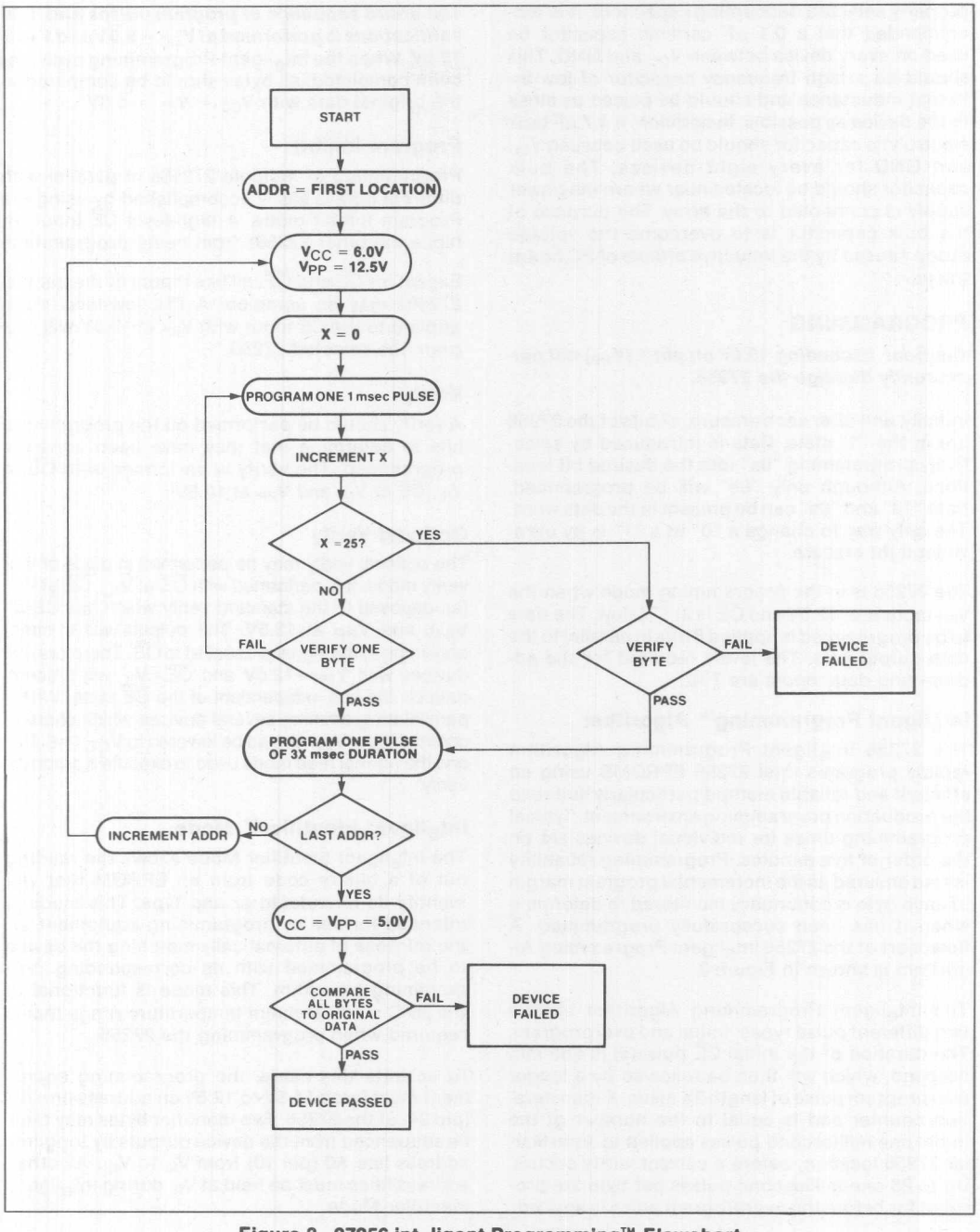


Figure 3. 27256 int$_e$ligent Programming™ Flowchart

Byte 0 (A0 = V_{IL}) represents the manufacturer code and byte 1 (A0 = V_{IH}) the device identifier code. For the Intel 27256, these two identifier bytes are given in Table 2. All identifiers for manufacturer and device codes will possess odd parity, with the MSB (O_7) defined as the parity bit.

ERASURE CHARACTERISTICS

The erasure characteristics of the 27256 are such that erasure begins to occur upon exposure to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000– 4000 Å range. Data show that constant exposure to room level fluorescent lighting could erase the typical 27256 in approximately 3 years, while it would take approximately 1 week to cause erasure when exposed to direct sunlight. If the 27256 is to be exposed to these types of lighting conditions for extended periods of time, opaque labels should be placed over the 27256 window to prevent unintentional erasure.

The recommended erasure procedure for the 27256 is exposure to shortwave ultraviolet light which has a wavelength of 2537 Angstroms (Å). The integrated dose (i.e., UV intensity × exposure time) for erasure should be a minimum of 15 Wsec/cm^2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with a 12000 μW/cm^2 power rating. The 27256 should be placed within 1 inch of the lamp tubes during erasure. The maximum integrated dose a 27256 can be exposed to without damage is 7258 Wsec/cm^2 (1 week @ 12000 μW/cm^2). Exposure of the 27256 to high intensity UV light for long periods may cause permanent damage.

Table 2. 27256 int$_e$ligent Identifier™ Bytes

Identifier \ Pins	A_0 (10)	O_7 (19)	O_6 (18)	O_5 (17)	O_4 (16)	O_3 (15)	O_2 (13)	O_1 (12)	O_0 (11)	Hex Data
Manufacturer Code	V_{IL}	1	0	0	0	1	0	0	1	89
Device Code	VIH	0	0	0	0	0	1	0	0	04

NOTES:
1. A_9 = 12.0V ± 0.5V
2. A_1–A_8, A_{10}–A_{13}, $\overline{CE}$, $\overline{OE}$ = V_{IL}
3. A_{14} = V_{IH} or V_{IL}

int$_e$ligent Programming™ Algorithm

D.C. PROGRAMMING CHARACTERISTICS:

T_A = 25 ± 5°C, V_{CC} = 6.0V ± 0.25V, V_{PP} = 12.5V ± 0.3V

Symbol	Parameter	Limits			Test Conditions (see Note 1)
		Min.	Max.	Unit	
I_{LI}	Input Current (All Inputs)		10	μA	V_{IN} = V_{IL} or V_{IH}
V_{IL}	Input Low Level (All Inputs)	−0.1	0.8	V	
V_{IH}	Input High Level	2.0	V_{CC}+1	V	
V_{OL}	Output Low Voltage During Verify		0.45	V	I_{OL} = 2.1 mA
V_{OH}	Output High Voltage During Verify	2.4		V	I_{OH} = −400 μA
I_{CC2}	V_{CC} Supply Current (Program & Verify)		100	mA	
I_{PP2}	V_{PP} Supply Current (Program)		50	mA	$\overline{CE}$ = V_{IL}
V_{ID}	A_9 int$_e$ligent Identifier Voltage	11.5	12.5	V	

NOTES:
1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.

A.C. PROGRAMMING CHARACTERISTICS:

$T_A = 25 \pm 5°C$, $V_{CC} = 6.0V \pm 0.25V$, $V_{PP} = 12.5V \pm 0.3V$

Symbol	Parameter	Limits				Test Conditions* (see Note 1)
		Min.	Typ.	Max.	Unit	
t_{AS}	Address Setup Time	2			μs	
t_{OES}	$\overline{OE}$ Setup Time	2			μs	
t_{DS}	Data Setup Time	2			μs	
t_{AH}	Address Hold Time	0			μs	
t_{DH}	Data Hold Time	2			μs	
t_{DFP}[4]	Output Enable to Output Float Delay	0		130	ns	
t_{VPS}	V_{PP} Setup Time	2			μs	
t_{VCS}	V_{CC} Setup Time	2			μs	
t_{PW}	$\overline{CE}$ Initial Program Pulse Width	0.95	1.0	1.05	ms	(see Note 3)
t_{OPW}	$\overline{CE}$ Overprogram Pulse Width	2.85		78.75	ms	(see Note 2)
t_{OE}	Data Valid from $\overline{OE}$			150	ns	

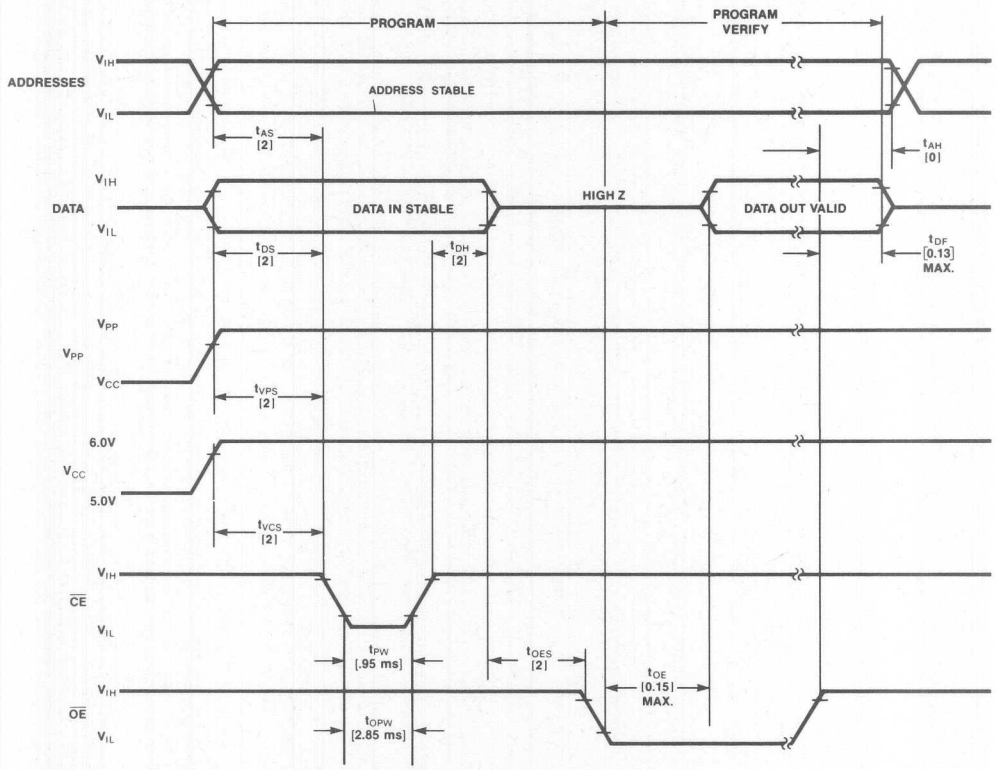
*A.C. CONDITIONS OF TEST

Input Rise and Fall Times (10% to 90%) . . . 20 ns
Input Pulse Levels . . . . . 0.45V to 2.4V
Input Timing Reference Level . . . . . 0.8V and 2.0V
Output Timing Reference Level . . . 0.8V and 2.0V

NOTES:

1. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.
2. The length of the overprogram pulse may vary from 2.85 msec to 78.75 msec as a function of the iteration counter value X.
3. Initial Program Pulse width tolerance is 1 msec ±5%.
4. This parameter is only sampled and is not 100% tested. Output Float is defined as the point where data is no longer driven—see timing diagram on the following page.

intelligent Programming™ WAVEFORMS



NOTES:
1. ALL TIMES SHOWN IN [ ] ARE MINIMUM AND IN μSEC UNLESS OTHERWISE SPECIFIED.
2. THE INPUT TIMING REFERENCE LEVEL IS .8V FOR A V_{IL} AND 2V FOR A V_{IH}.
3. t_{OE} AND t_{DFP} ARE CHARACTERISTICS OF THE DEVICE BUT MUST BE ACCOMODATED BY THE PROGRAMMER.
4. WHEN PROGRAMMING THE 27256 A 0.1μF CAPACITOR IS REQUIRED ACROSS V_{PP} AND GROUND TO SUPRESS SPURIOUS VOLTAGE TRANSIENTS WHICH CAN DAMAGE THE DEVICE.

NVRAM (Non-Volatile Random Access Memories)

3

2004
4K (512 x 8) NON-VOLATILE RANDOM ACCESS MEMORY

- 5 Volt Only Operation
- Fast Static RAM Read/Write Cycles
  2004-2, 200ns MAX.
  2004, 250ns MAX.
  2004-3, 300ns MAX.
- Single Line STORE & RECALL
- 10ms Self-Timed STORE Cycles for 2004-2 and 2004 (20ms for 2004-3)
- Automatic Recall on Power Up
- Write Protect Circuit to Preserve Data On Power-Up and Power-Down
- Lower Power Standby Mode
- 10-Year Data Retention for each STORE
- Minimum 10,000 Non-Volatile STORE Cycle Endurance
- Unlimited Endurance for Read, Write, and RECALL Cycles
- HMOS-E FLOTOX Cell Design
- Conforms to JEDEC Byte-Wide Universal Site

The Intel 2004 Non-Volatile Random Access Memory (NVRAM) is a 4K device with 512 × 8 architecture. It provides the real-time read/write functions of a static RAM together with the reliable non-volatile storage capability of an E²PROM array to preserve its memory contents when power is removed.

Internally, the 2004 NVRAM consists of a high speed static RAM array backed up, bit-for-bit, by an E²PROM array for non-volatile storage. The transfer of memory data between the static RAM and the E²PROM array occurs in parallel for fast storage and recall as well as minimal system support.

Two functions are provided to transfer data between the volatile RAM and its non-volatile E²PROM counterpart. The STORE function transfers RAM data into the E²PROM while the RECALL function fetches E²PROM data and places it in the RAM array. Both functions are controlled by a single $\overline{NE}$ signal which can easily be activated with traditional circuitry in memory mapped space, through an I/O port, or from the output of a power-fail detector.

The RAM operating characteristics of the 2004 NVRAM provides high speed microprocessor performance with unlimited endurance. In the non-volatile storage mode, data retention is specified at over 10 years for each STORE operation. Over 10,000 STORE operations can be performed reliably.

The 2004 NVRAM is furnished in a 28-pin byte wide package with its address, data and control lines configured according to the standard JEDEC universal 28-pin site.

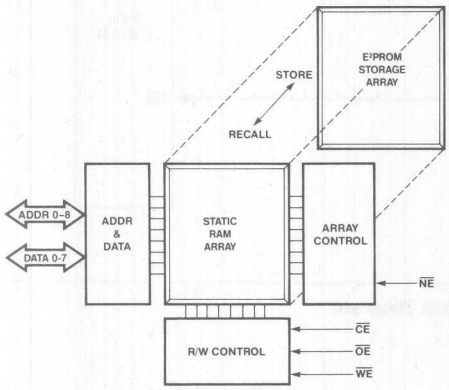


Figure 1. 2004 Functional Diagram

2004 PIN NAMES

A_0-A_8	ADDRESSES
I/O	DATA INPUT/OUTPUTS
$\overline{CE}$	CHIP ENABLE
$\overline{OE}$	OUTPUT ENABLE (READ)
$\overline{NE}$	NON-VOLATILE ENABLE (STORE/RECALL CONTROL)
$\overline{WE}$	WRITE ENABLE
N. C.	NO CONNECT

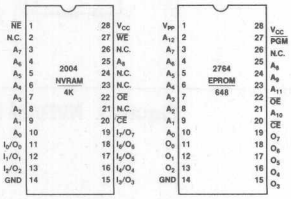


Figure 2. 2004 Pin Configuration

JUNE 1983

ORDER NUMBER: 230665-002

NVRAM APPLICATIONS

The non-volatile RAM is designed for operation in systems where important variable data needs to be retained during periods when power is not applied to the system. It combines the flexibility of a static RAM with the reliable non-volatility of E²PROM.

The NVRAM provides a more flexible alternative than the E²PROM. E²PROMs are best suited for non-volatile storage of program code or system parameters which are occasionally altered. The NVRAM is designed for fast non-volatile storage of multiple data bytes which are in transit in the system or are being altered at microprocessor speeds.

Both NVRAMs and E²PROMs provide important, but separate, functions in the same system. An example is in communication equipment. The NVRAM is used for buffer storage for data being received or transmitted over a communications link. If power fails, a single microprocessor instruction causes the NVRAM to store all the buffered data. The E²PROM array in this system is used to store the microprocessor program code as well as look-up tables for the various operating parameters. The E²PROM program code can be updated via the communications link, and the operating parameters can be modified either remotely over the link or directly from the keyboard.

Non-volatile RAMs offer a silicon alternative to memory designs using battery-back CMOS RAM. The simple operation, single chip solution, and superior reliability of the NVRAM is a preferable alternative to the chemical battery. The NVRAM's reliable non-volatile storage is designed for a minimum of 10,000 store cycles, each with a data retention of 10 years.

The 2004 non-volatile RAM features 2-line control. By requiring a chip enable ($\overline{CE} = 0$) whenever a device function is to be activated (determined by $\overline{OE}$, $\overline{WE}$, and $\overline{NE}$), data bus contention is eliminated, system noise is reduced, and system design is simplified.

SYSTEM CONSIDERATIONS

Figure 3 illustrates a typical hardware system's block diagram. A power fail detection circuit is used to notify the system CPU of the loss of AC power via an interrupt line. The microprocessor decodes the interrupt, enters a servicing routine which writes important system data into the NVRAM, then performs a STORE initiation cycle. The STORE operation is completed in 10ms during which the power supply has held Vcc at 5V. As power is finally lost in the system, a Vcc STORE protection circuit prevents any subsequent unwanted STORE operations from being started due to system power-down noise. Data is retained in the non-volatile storage array in the NVRAM.

Figure 4 shows how to generate the $\overline{NE}$ signal with an I/O port bit for an array of 2004s.

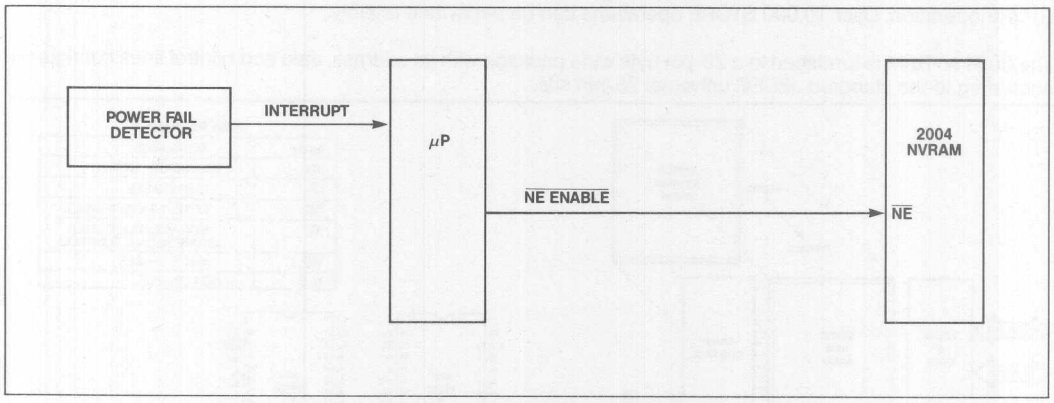


Figure 3. NVRAM System Block Diagram

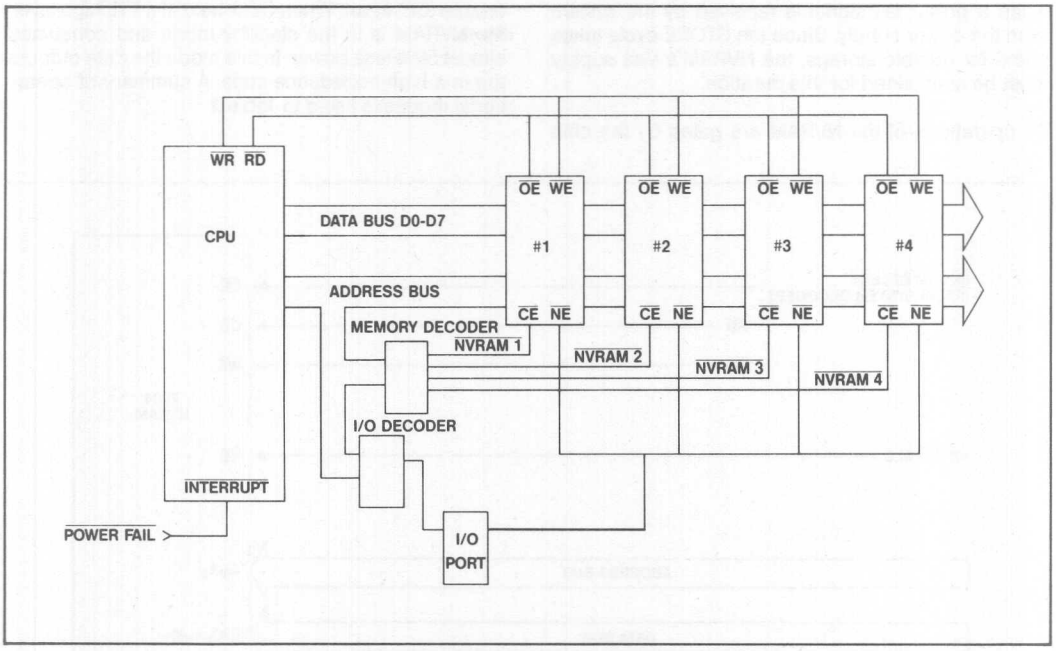


Figure 4. 2004 Array: Generating $\overline{NE}$ by I/O Output Port Bit

OPERATIONAL OVERVIEW

The non-volatile RAM uses standard control lines in accordance with established microprocessor interfaces and pinout standards. An additional control pin, non-volatile enable ($\overline{NE}$), is used to control the STORE and RECALL functions to the non-volatile storage array. For RAM access cycles the $\overline{NE}$ input is held high on the NVRAM. Addressing of each of the individual 512 RAM array bytes is accomplished using address lines 0 through 8 (A0-A8) and enabling the chip ($\overline{CE}$). Data is then transferred a byte (8 bits) at a time to and from the RAM array in the NVRAM. Data is written by lowering Write Enable ($\overline{WE}$) and holding Output Enable ($\overline{OE}$) at a TTL high level. Data is read by lowering the $\overline{OE}$ input while holding the $\overline{WE}$ input high. These functions are generally accomplished using a hardware design as shown in Figure 5.

The non-volatile functions of the NVRAM are implemented by putting a TTL low on the $\overline{NE}$ pin. A RECALL operation is initiated by bringing $\overline{OE}$ low while $\overline{NE} = 0$. This causes the data from the NVRAM's internal non-volatile storage array to be transferred to the static

RAM array, from which it can be externally accessed. The RECALL function can be activated by the CPU at any time, as often as desired without affecting the integrity of the data in the non-volatile storage array. The RECALL function is also automatically activated when the NVRAM is powered up.

A STORE operation is started by bringing $\overline{WE}$ low while holding $\overline{NE}$ low. This causes the data in the static RAM array to be transferred to the non-volatile storage array. The STORE function is typically used to save data during power failure periods and is executed

when a power fail signal is received by the system from the power supply. Since the STORE cycle takes 10ms for reliable storage, the NVRAM's Vcc supply must be maintained for this duration.

All operations of the NVRAM are gated by the chip enable ($\overline{CE}$) input. When $\overline{CE}$ is held at a TTL high level, the NVRAM is in the standby mode and consumes almost 50% less power. In this mode the data outputs are in a high impedance state. A summary of operational modes is listed in Table 1.

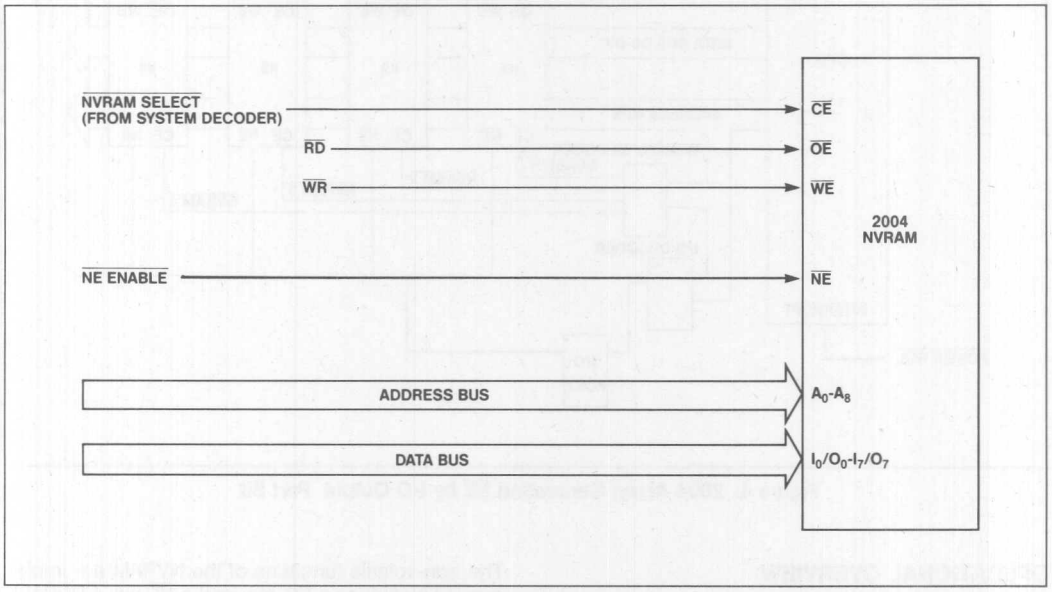


Figure 5. NVRAM System Interface

INADVERTENT STORE PROTECTION DURING POWER FAIL

When system power is falling the TTL devices which generate the $\overline{CE}$, $\overline{OE}$, $\overline{WE}$, and $\overline{NE}$ signals will no longer be stable. There is a possibility that the noise on these lines could initiate a STORE operation as Vcc is falling. Either an on-chip circuit or protection circuit with a lower operating voltage than the rest of the system is needed to prevent an inadvertent STORE operation. The 2004 has an on-chip inadvertent write protection circuit. When Vcc is below 3.5V, the device's write mode is disabled. If the surrounding system is stable when Vcc is above 3.5V, then the on-chip circuit will be sufficient protection against a spurious write. If the system is still unstable when Vcc is above 3.5V, an external circuit such as the one in Figure 6 can be used. An example of such a circuit is shown in Figure 6.

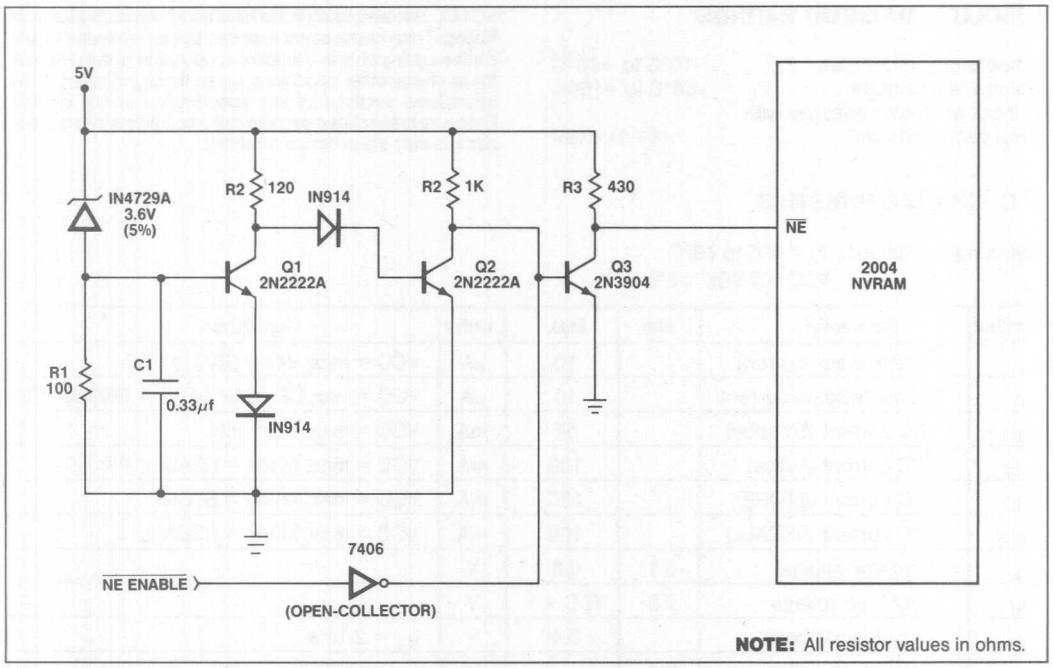


Figure 6. External Circuit Example for Inadvertent STORE Protection During Power-Up and Power-Down

The $\overline{NE}$ signal from the microprocessor (or via an I/O port) is gated to the NVRAM through the circuit shown in Figure 6. This circuit allows the $\overline{NE}$ signal through to the NVRAM as long as Vcc is above 4V. Should Vcc fall below 4V, transistor Q1 turns off, forcing transistor Q2 to conduct and effectively grounding the base of Q3. Q3 then turns off, holding the NVRAM $\overline{NE}$ input at the Vcc level through resistor R3. The $\overline{NE}$ input thus remains disabled as Vcc falls to 0V.

Table 1. 2004 Operational Modes Vcc = 5V

	$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	$\overline{NE}$	Outputs
Standby	V_{IH}	X	X	X	Hi–Z
READ	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Data Out
WRITE	V_{IL}	X	V_{IL}	V_{IH}	Data In
RECALL–Power Up	X	X	X	V_{IH}	Hi–Z
RECALL–Standard	V_{IL}	V_{IL}	V_{IH}	V_{IL}	Hi–Z
STORE	V_{IL}	V_{IH}	V_{IL}	V_{IL}	Hi–Z

ABSOLUTE MAXIMUM RATINGS*

Temperature Under Bias . . . . . -10°C to +80°C
Storage Temperature . . . . . -65°C to +100C
All Input or Output Voltages with
Respect to Ground . . . . . +6V to -0.3V

*NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D. C. CHARACTERISTICS

Operating Conditions: T_A = 0°C to 70°C
VCC = 5 VDC ± 5%

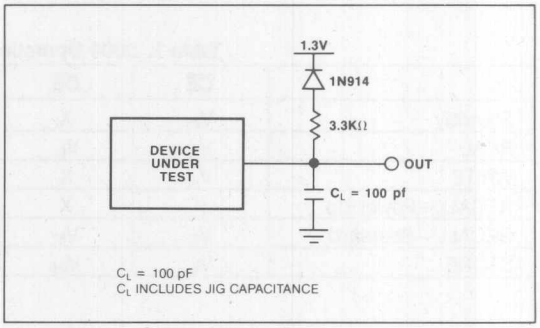
Symbol	Parameter	Min	Max	Units	Conditions
I_{LI}	Input leakage current		10	μA	VCC = max, VIN = GND to VCC
I_{LO}	Output leakage current		10	μA	VCC = max, $\overline{CE}$ = VIH, VOUT = GND to VCC
I_{CC1}	VCC current (Standby)		55	mA	VCC = max, $\overline{CE}$ = VIH
I_{CC2}	VCC current (Active)		100	mA	VCC = max, Mode = READ or WRITE
I_{CC3}	VCC current (STORE)		100	mA	VCC = max, Mode = STORE
I_{CC4}	VCC current (RECALL)		100	mA	VCC = max, Mode = RECALL
V_{IL}	Input low voltage	-0.1[1]	0.8	V	
V_{IH}	Input high voltage	2.0	VCC + 1	V	
V_{OL}	Output low voltage		0.4	V	I_{OL} = 2.1ma
V_{OH}	Output high voltage	2.4		V	I_{OH} = -400μa
V_{RCL}	V_{CC} level at which automatic RECALL begins during Power-Up	3.0	3.5	V	

Note 1. -1.0V spikes less than 20ns in duration are allowed

A. C. Test Conditions

Input pulse levels: 0.45V and 2.4V
Input rise and fail times: 20 nsec (10% and 90% levels)
Output timing reference levels: 0.8V and 2.0V

A.C. TESTING LOAD CIRCUIT



A. C. CHARACTERISTICS

READ CYCLE

Symbol	Parameter	2004-2		2004		2004-3		Units
		Min	Max	Min	Max	Min	Max	
t_{RC}	Read Cycle Time	200		250		300		ns
t_{ACC}	Address Access Time		200		250		300	ns
t_{CE}	Chip Select Access		200		250		300	ns
t_{OE}	$\overline{OE}$ Access Time		70		100		150	ns
t_{CLZ}	$\overline{CE}$ Selection to Active Output	10		10		10		ns
t_{CHZ}	$\overline{CE}$ Deselection to Output Not Driven		60		60		130	ns
t_{OLZ}	$\overline{OE}$ Selection to Active Output	10		10		10		ns
t_{OHZ}	$\overline{OE}$ Deselection to Output Not Driven		60		60		130	ns
t_{PU}	$\overline{CE}$ Selection to Power Up Time[1]	10		10		10		ns
t_{PD}	$\overline{CE}$ Deselection to Power Down Time[1]		90		120		160	ns
t_{OH}	Output Held from Addresses, $\overline{CE}$, or $\overline{OE}$, (whichever occurs first)	0		0		0		ns

WRITE CYCLE

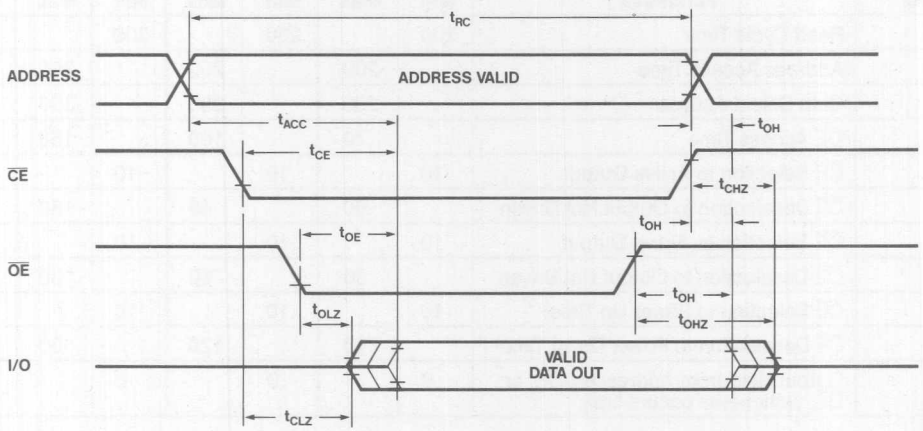
Symbol	Parameter	2004-2		2004		2004-3		Units
		Min	Max	Min	Max	Min	Max	
t_{WC}	Write Cycle Time	200		250		300		ns
t_{CW}	Chip Selection to End of Write	200		250		300		ns
t_{AW}	Address Valid to End of Write	200		250		300		ns
t_{AS}	Address Setup Time	0		0		0		ns
t_{WP}	Write Pulse Width	120		150		200		ns
t_{DW}	Data Valid to End of Write	120		150		200		ns
t_{DH}	Data Hold Time	0		0		0		ns
t_{WLZ}	$\overline{WE}$ Disabled to Active Output	10		10		10		ns
t_{WHZ}	$\overline{WE}$ Enabled to Output Not Driven		60		90		130	ns
t_{WR}	Write Recovery Time	0		0		0		ns

Note 1. This refers to the powering up and down of the device's internal circuitry.

WAVEFORMS

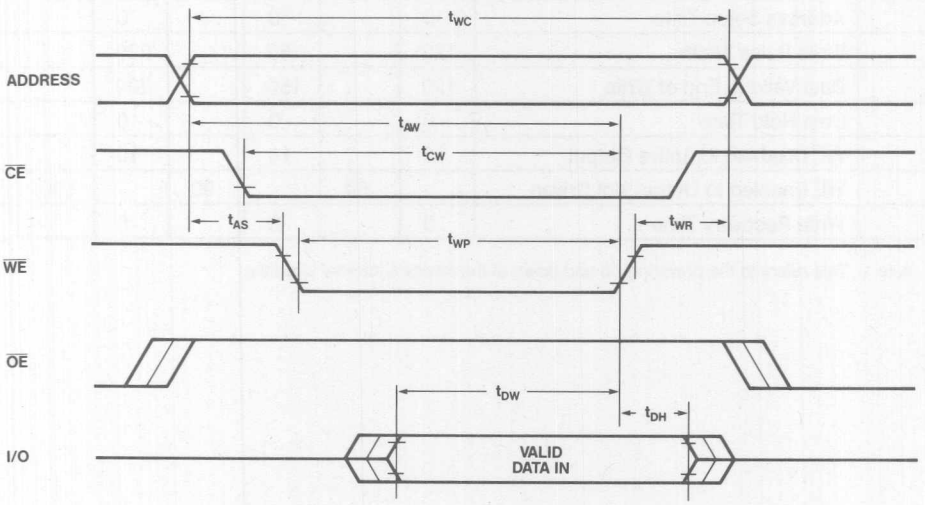
READ

CONDITIONS: $\overline{NE} = V_{IH}$, $\overline{WE} = V_{IH}$



WRITE

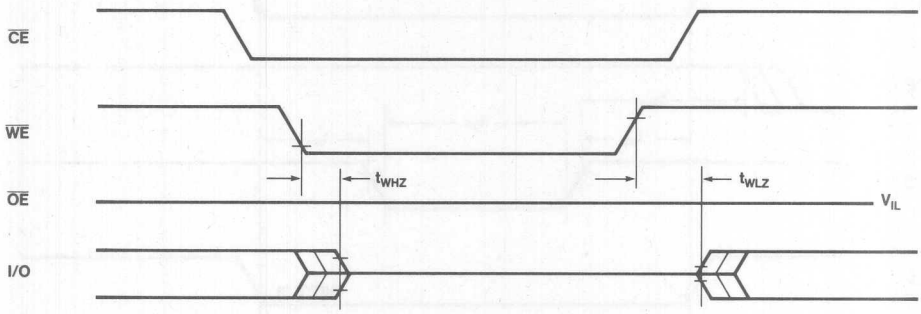
FOR SYSTEMS WITH 2-LINE CONTROL (INTEL MICROPROCESSORS)
CONDITION: $\overline{NE} = 6$



WAVEFORMS (CONT.)

WRITE

WITH SINGLE LINE CONTROL
CONDITION: $\overline{NE} = V_{IH}$ (OUTPUT ENABLE AND DISABLE TIMES SHOWN ONLY)

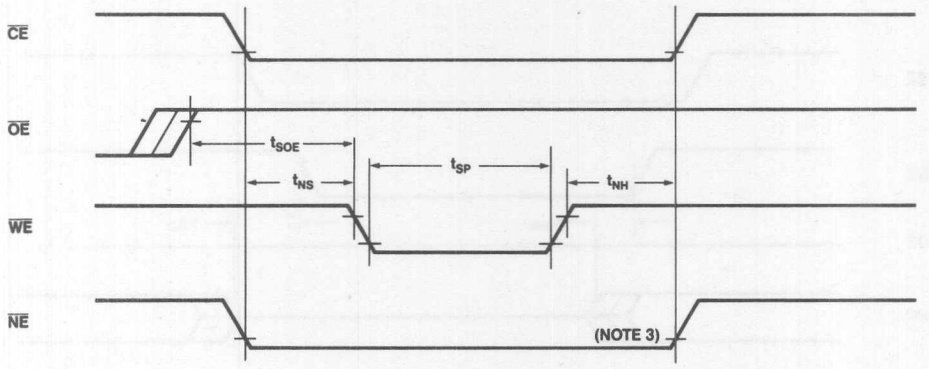


STORE Operation[1, 2]

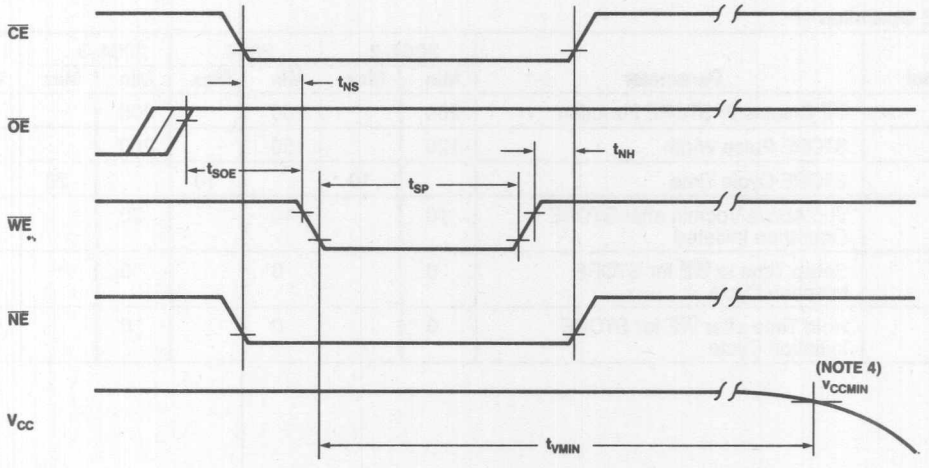
Symbol	Parameter	2004-2		2004		2004-3		Units
		Min	Max	Min	Max	Min	Max	
t_{SOE}	$\overline{OE}$ Disable to STORE Function	200		200		200		ns
t_{SP}	STORE Pulse Width	120		150		200		ns
t_{STR}	STORE Cycle Time		10		10		20	ms
t_{VMIN}	Vcc Above Vccmin after STORE Operation Initiated	10		10		20		ms
t_{NS}	Setup Time to $\overline{WE}$ for STORE Initiation Cycle	0		0		0		ns
t_{NH}	Hold Time after $\overline{WE}$ for STORE Initiation Cycle	0		0		0		ns

WAVEFORMS (CONT.)

STORE INITIATION CYCLE UNDER NORMAL CONDITIONS



STORE INITIATION CYCLE DURING POWER FAIL



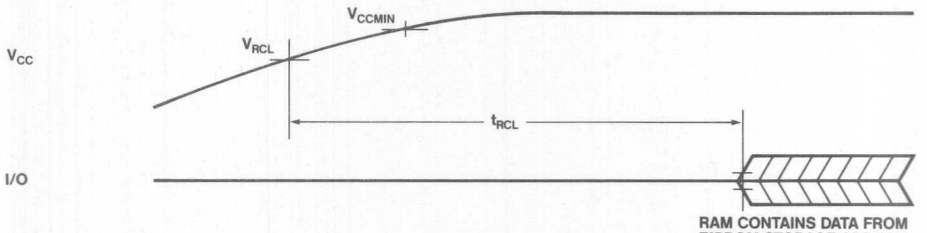
NOTES:

1. During a STORE initiation cycle, addresses and I/O inputs = DON'T CARE.
2. For a STORE cycle, $\overline{WE}$ must not go low before $\overline{NE}$. Otherwise, a RAM write cycle will result.
3. A lockout feature on the $\overline{NE}$ input prevents subsequent STORE cycles from occurring until $\overline{NE}$ is brought back high. The $\overline{NE}$ input should therefore be brought back high after a non power-fail STORE operation is initiated to allow normal read/write access after completion of the STORE operation.
4. Vccmin = 4.75V for 5% Vcc spec
5. Once a STORE operation has begun, all inputs are ignored and the outputs are in a high impedance state.

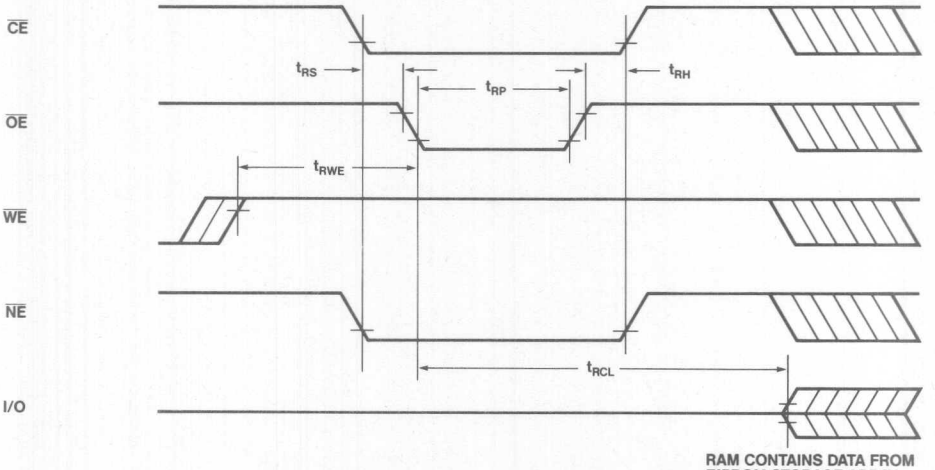
RECALL Operation[1]

Symbol	Parameter	2004-2		2004		2004-3		Units
		Min	Max	Min	Max	Min	Max	
t_{RWE}	$\overline{WE}$ Disable to RECALL Command	200		200		200		ns
t_{RP}	RECALL Pulse Width	120		150		200		ns
t_{RCL}	RECALL Cycle Time		10		10		10	μs
t_{RS}	Setup Time to RECALL Command	0		0		0		ns
t_{RH}	Hold Time after RECALL Command	0		0		0		ns

AUTOMATIC RECALL DURING POWER UP[2]



RECALL INITIATION CYCLE UNDER NORMAL CONDITIONS (NOTE 3)



NOTES:

1. During a RECALL Initiation Cycle, the address and data inputs = DON'T CARE.
2. During Automatic Power-Up RECALL, $\overline{CE}$ = $\overline{WE}$ = $\overline{OE}$ = DON'T CARE.
3. Once a RECALL operation has begun, all inputs are ignored and the outputs are in a high impedance state.
4. Vccmin = 4.75V for 5% Vcc spec

Bubble Memory

4

April 1983

New Bubble-Memory Packaging Cuts Board Space and Manufacturing Costs

Art Thorp
Intel Corp.
Santa Clara, Calif.

ORDER NUMBER: 230642-001

New bubble-memory packaging cuts board space and manufacturing costs

Low-profile 4-Mb bubble-memory package is interchangeable with 1-Mb types and also lets printed-circuit boards be spaced on 0.6-in. centers

by Art Thorp, *Intel Corp., Santa Clara, Calif.*

☐ Designing a second-generation product gives an engineering team the chance to put in all the improvements they realized were needed after the first design was formalized. The new 7114 4-megabit magnetic-bubble memory from Intel makes the most of this opportunity in terms of its ease of both use and manufacturing.

Despite the quadrupled bit density, the 7114's leaded package is smaller in all three dimensions than the leadless package of its 1-Mb predecessor, the 7110. It occupies less space on a printed-circuit board and has a lower profile—low enough for the boards carrying it to fit into adjacent rather than alternate slots in standard card cages. Moreover, chip and package are far easier and cheaper to assemble.

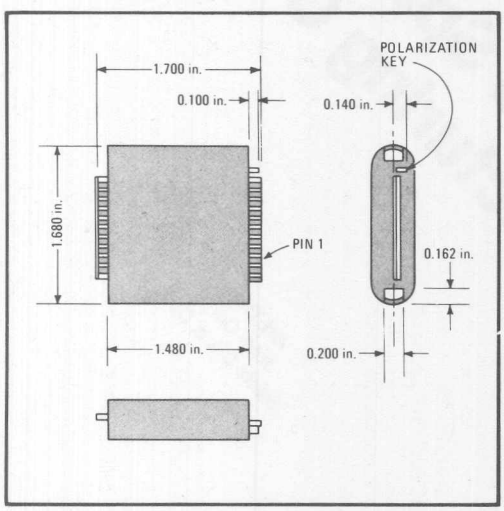
Nor is that convenience compromised by a lack of compatibility with the 7110. The pinouts are the same, and the pin spacings sufficiently similar to make it simple to upgrade from the 7110 to 7114. Also, the support circuits essential to the control of each bubble memory are either identical or so alike as to be interchangeable.

More specifically, the first-generation 1-Mb bubble device is a 520-by-620-mil chip in a leadless package that needs a socket; the assemblage has a footprint of 2.20 by 1.825 in. (Fig. 1) and an overall height of 0.430 in.

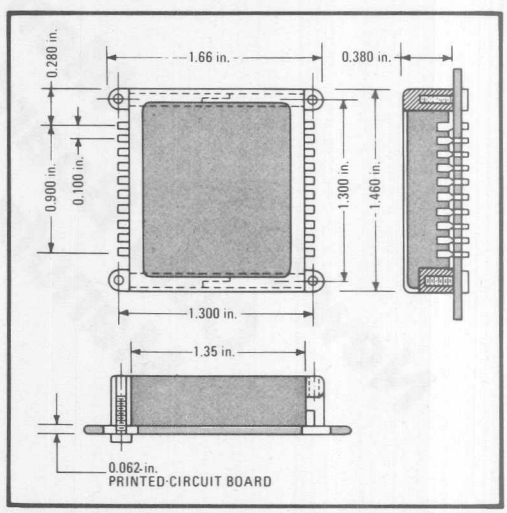
In contrast, the 4-Mb chip and a forthcoming 1-Mb device are smaller—580 by 500 mils—and their leaded "thin-C" dual in-line package has a footprint of only 1.66 by 1.46 in. (Fig. 2). Also, its height is now only 0.375 in., so that when inserted either directly into a pc board or in a zero-profile socket, it leaves ample clearance for the 0.6-in. card spacing normal in commercial card cages.

Design goals

Without scaling down device geometries, it would have been impossible to fit a garnet chip containing four times as many bubble domains into a package of the same size, let alone a smaller one. Thus the first order of business was at least to halve device geometries in both dimen-



1. Leadless. The leadless package of this first-generation 1-megabit magnetic-bubble memory has a footprint of 2.20 by l.825 inches and an overall height of 0.430 in. when socketed. Thus boards cannot be spaced the standard 0.6 in. apart.



2. Thin and leaded. Despite its 4-Mb capacity, this bubble chip fits in a leaded package with only a 1.66-by-1.46 in. footprint and 0.375-in. profile. Cards carrying these packages or using zero-profile sockets for them can be set into a card cage with the standard 0.6-in. spacing.

sions. The production application of X-ray lithography, in fact, yields 4-Mb bubble chips that are smaller than the 1-Mb one in the current 7110 leadless package.

In addition to different die dimensions, the smaller package required smaller magnets and coils with different dimensions to help control the flow of magnetic bubbles on the garnet chip. A program for designing models of coil size and shape was therefore developed and run on an IBM Personal Computer.

Either the 1- or 4-Mb scaled-down bubble device could have been placed in the same package as the original 1-Mb memory if that had been desired. Instead, it was decided not to settle for the existing package but to produce a new one that, while compatible with the 7110, would be more useful to the engineer—namely, by being smaller and allowing standard pc board spacing. Equally important, if not more so, was the decision to make the production process more efficient and cost-effective.

Nevertheless, there were to be no compromises in the stiff specifications for durability, magnetic shielding, and temperature range. In essence, the design goal was for the new package to be at least as good as the first in some aspects and better in others.

A thinsy

One major concern in moving to a new package is its effect on users who are already producing systems containing the first-generation version and who plan to continue manufacturing while introducing the later one. Unless pinout and spacings are absolutely identical, the transition to a new package cannot be totally painless.

However, in this case, maintaining the identical spacings both within and between the two rows of pins would eliminate any benefit gained by a smaller package. Thus, the decision was to keep the 7114's pinout and adjacent pin spacings the same as on the leadless 7110 package. Only the separation between the two rows of pins has been made smaller on the new memories.

As a result those engineers now manufacturing equipment using the previous 7110 model can lay out their pc boards in such a manner as to accommodate either the first-generation package or, with a minimal amount of revision, the new one. The trick is to elongate and drill the pin trace pads on the board for two holes per pin, as shown in Fig 3. For new layouts, this scheme should be used from the very start. Existing system boards can be modified this way with little effort. If board-level diagnostic and maintenance operations require the use of sockets for the packages, the pc-board holes should be dimensioned for the zero-profile–socket contacts, such as Augat Holtite types. Otherwise, the advantage of the 0.6-in. board spacing will be lost.

Attacking manufacturing costs

In many ways, making magnetic-bubble chips is similar to making integrated circuits, but there are significant differences—for instance, semiconductors do not need wire coils. Therefore, it is reasonable to assume that the major manufacturing cost factors in the one process will not be the same for the other.

In the original 7110, the garnet bubble chip is first die-bonded to a ceramic substrate and then wire-bonded to conductors metalized onto the ceramic. Next, the field and drive coils are put in place around the garnet chip, and all the components are potted using a liquid epoxy compound. Both the use of ceramic and the potting process contribute heavily to memory cost.

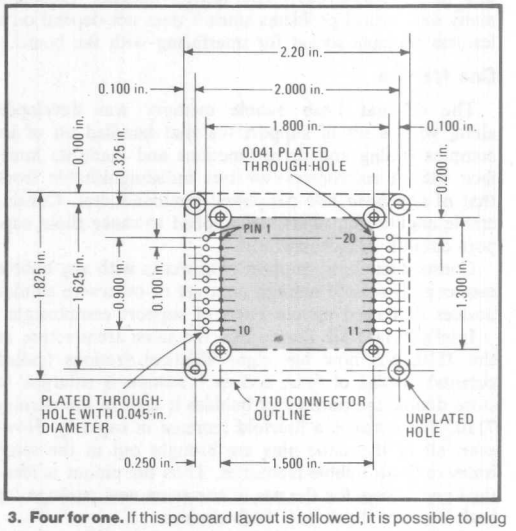
The 7114 package design is more economical on both counts. For the ceramic substrate, the design team substituted a special pc-board material that is thinner, lighter in weight, and lower-cost.

Next, the designers tackled the problem of potting the chip, substrate, and coil combination. Packages containing ICs often employ transfer molding of a thermoset epoxy compound. But when applied to the bubble assembly, the high pressures involved in this process—about 500 to 1,000 pounds per square inch—routinely deformed the bubble memory's wire coils and degraded their electrical performance unacceptably.

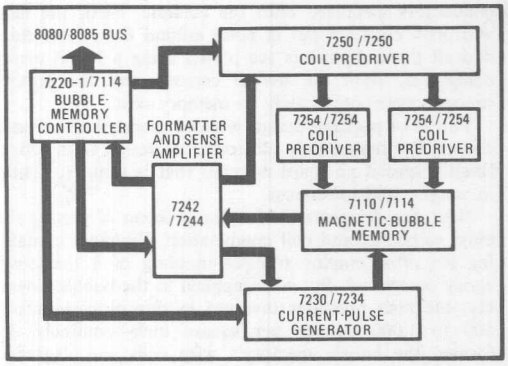
Indeed, for the 7110, manual potting had seemed unavoidable, even though production personnel spent an average of 1 hour on each bubble assemblage, first placing it in a mold, then pouring in liquid epoxy, placing it in a high-temperature oven to cure, removing the mold from the oven, and finally extracting the assembly.

Nonetheless, for the 7114 a fast alternative was found in the liquid injection-molding process used by some manufacturers for high-voltage insulators. Unlike transfer molding, it employs only low pressure, in the region of 15 psi, and it speeds up the potting process to more than 50 devices per hour.

Thus by replacing the ceramic substrate with pc-board material and by substituting liquid injection molding for the manual pouring of a potting compound, the package engineering team had a very favorable impact on the cost and throughput of manufacturing. An added advantage is that, with the ceramic removed, the possibility of chip-



3. Four for one. If this pc board layout is followed, it is possible to plug either a socketed 7110, a leaded 4-Mb, or a leaded 1-Mb bubble-memory package into this hole pattern. The socketed 7110 goes into the outer set of holes, while the newer packages fit the inner set.



4. Upgrade. Along with the bubble chip, all the other ICs of the 1- and 4-Mb memory systems are mechanically and electrically interchangeable so far as the pc board layout is concerned. The diagram shows the chip set for a 4-Mb system in color and the 1-Mb system in black.

ping the exposed edges has been reduced. The table on this page summarizes the packaging and manufacturing aspects of the new and old bubble package methods.

There is always the danger that improving one aspect of a product may inadvertently degrade another. In this case, however, that has not been the result. For example, the original 7110 package was designed to offer protection from external magnetic fields to a level of 20 oersteds. Furthermore, the 7110 is specified to operate over a standard temperature range of 0° to 75°C.

In each case, the new package offers the same or better specifications than the leadless package. In terms of mechanical reliability, both the leadless and leaded package meet and exceed all vibration and shock test limits specified in MIL-STD-883. The leaded package precludes many mechanical problems since it does not depend on a leadless package socket for interfacing with the board.

One for one

The original 1-Mb bubble memory was developed along with a set of support ICs that handled all of its complex timing and drive functions and made its interface with a microprocessor bus indistinguishable from that of any *bona fide* peripheral semiconductor. Considerable engineering effort was applied to make these support circuits interchangeable.

Consequently, any support chip works with any bubble memory. This is in marked contrast to otherwise similar devices that need matched sets of support components.

Intel's 7114 4-Mb device uses the same architecture as the 7110, but now has eight identical sections (called octants) instead of four, and each section is enlarged to store double the number of bubbles it does in the current 7110. The result is a fourfold increase in capacity. However, all of the same pins are brought out in the same order on both bubble memories. Thus the pinout is identical and allows for the use of the same new package for both the new 1-Mb and 4-Mb devices.

Those support ICs that are not affected by the increased capacity, such as the coil predriver (7250) and drivers (7254), are used with either memory.

FIRST- VERSUS SECOND-GENERATION MAGNETIC-BUBBLE MEMORIES

Type	7110 Leadless	7110 Leaded Thin C	7114 Leaded Thin C
Memory (Mb)	1	1	4
Die size (in.)	0.620 by 0.520	0.580 by 0.500	0.580 by 0.500
Substrate material	ceramic	printed-circuit board	printed-circuit board
Potting technique	liquid epoxy by hand	liquid injection molding	liquid injection molding

Those support ICs that have been designed in conjunction with the new memory are the same size and have the same pinouts as their counterparts in the 7110 1-Mb subsystems. What has changed is some of the programmable parameters and the descriptions of their associated registers. These are all involved with the new bubble-memory controller chip—the IC that interfaces the microprocessor bus with the 4-Mb memory.

Examining the effect of these changes in upgrading from a 7110 1-Mb to a 7114 4-Mb package reveals that the modifications are really minimal—the new support ICs can be designed into the same board layouts as their 1-Mb cousins. Users will have only to make some modifications in their software to handle minor differences in addressing and configuration initialization.

Intel's bubble-memory system therefore can have the same configuration whether working with 1-Mb or 4-Mb devices. A single bubble-memory controller acts as the interface between the microprocssor bus and one or more bubble storage subsystems.

The 7224 controller for the 4-Mb bubble-memory device is housed in a standard 40-pin, dual in-line package and takes up about 2 by 0.5 in. on a board. A single controller operates up to eight storage subsystems for a maximum capacity of 4 megabytes.

Each bubble-memory subsystem contains a monolithic formatter and sense amplifier in a standard 20-pin DIP; a current-pulse–generator chip in a 22-pin DIP; a coil-predriver chip in a 16-pin DIP; a pair of quad V-groove MOS driver chips, each in a 14-pin DIP; and of course the bubble device itself. One subsystem takes up less than 3 by 4 in., and a 4-megabyte board of eight of them plus a a controller could be constrained to 6.75 by 12 in.

Obviously, boards laid out for the 7110 1-Mb memory and its support family would be approximately the same size for one fourth the amount of memory. However, in a card cage with a standard 0.6-in. spacing, boards built using the original leadless packages could not be stacked in adjacent slots.

Converting from the earlier 7110 1-Mb to a 4-Mb system essentially requires modifications to the pin pads of the 7110's leadless package. The 7220-1 bubble-memory controller is the same size and has the same pinout as the 7224. The same is true for the 7242 formatter and sense amplifier and its 7244 replacement, as well as for the 7230 current-pulse generator and its 7234 substitute. Figure 4 shows how an identical board can support either a 7114 4-Mb or a leaded or leadless 7110 1-Mb system.

Space limitations, interface details, and other such criteria will typically dictate the actual board layout. □

June 1983

Software Design and Implementation Details for Bubble Memory Systems

Richard Pierce
Applications Engineer
Intel Corporation

ORDER NUMBER: 230707-001

INTRODUCTION

The 7220-1 is a single-chip LSI Bubble Memory Controller (BMC) that implements a bubble memory storage subsystem (with up to eight bubble storage units (BSUs) per BMC). Each bubble storage unit consists of five support circuits in addition to the bubble memory chip and provides one megabit (128 kbytes) of non-volatile read/write memory. This application note examines the programmatic interface to the 7220-1 BMC and how communications between a host processor and the bubble subsystem (i.e., 7220-1) govern all bubble system operations.

The BMC provides all the control and timing signals for the bubble and support circuits. All data synchronization and error checking is automatically performed by the bubble subsystem to ensure reliable data storage. The BMC easily interfaces to microprocessor systems and communicates via a set of high-level commands. This application note explains the operations and functions performed by these instructions and provides the basic programmatic interface descriptions and program guidelines to allow you to design and implement a program module or "driver" to control all bubble system operations. In addition, several possible software interface levels are defined. While the design guidelines presented are not targeted for integration with any particular operating system, they do, however, provide conceptual information on design requirements and serve as a foundation on which to develop a modular and flexible software driver aligned with your specific application requirements.

Product Line Overview

Intel offers a complete line of bubble memory components, development kits and assembled boards.

The BPK 72 (Bubble Memory Prototype Kit) serves primarily as a means to evaluate the potential of bubble storage. The BPK 72 comes complete with all the hardware and documentation necessary to prototype a one-megabit bubble memory system. After the kit is assembled, the designer is left with the simple task of interfacing to a host processor.

The BPK 70 one-megabit bubble storage subsystem is a fully interchangable component bubble memory system. Each kit contains all the components in a Bubble Storage Unit (BSU). A single 7220-1 Bubble Memory Controller (purchased separately) can operate up to eight BPK 70 subsystems at one time. The BPK 70 is available for the production of custom systems where high volume is required.

The iSBX™ 251 Magnetic Bubble MULTIMODULE™ board is a one-megabit bubble memory mounted on a standard dual width Intel MULTIMODULE memory expansion card. Completely assembled and tested, the iSBX-251 board is fully plug compatible with all Intel iSBC Single Board Computers that have iSBX connectors.

The iSBC® 254 Bubble Memory Board is a completely assembled Intel MULTIBUS® memory board. The board can be configured with one bubble memory (128 kbytes), two bubble memories (256 kbytes), or four bubble memories (512 kbytes).

The 7220-1 BMC acts as the interface to the host processor in each of the aforementioned products, thus simplifying system programming. The basic programming techniques discussed in this application note will provide the system programmer with a complete understanding of programming requirements and shorten the software development time.

SOFTWARE INTERFACE

Basic Driver Operation

As will become evident, a basic compromise or "tradeoff" exists between the design of your application software and the capabilities of the bubble memory controller. While you will be responsible for the development of a driver to integrate the bubble into your system, the level of driver interaction and degree of flexibility will vary according to the needs of your application. If an application program is small and simple, a basic bubble driver simply may be called from the main program. At the next level of driver sophistication, the bubble system is viewed by the application program as a logical device. At this level, the key to driver design is the mapping of the "logical bubble interface" (as viewed by the application program) into the "physical bubble interface" (as implemented by the bubble drivers). This logical-to-physical mapping serves to isolate application programs and system software from the idiosyncracies of the bubble memory controller. At the

highest level of driver sophistication, the application program treats the bubble system as a collection of named data areas or files similar to the way in which data is stored and retrieved in disk operating systems. At the file system level, an application program can ignore the mechanics of bubble storage and access and merely present a "file name" to the driver to open, read or write, and then close the desired "bubble file."

At the subsystem level (i.e., "physical bubble interface"), the bubble driver is responsible for all system interaction with the bubble and is intrinsic to the efficient and reliable operation of the bubble system. The driver accepts bubble memory commands and command execution parameters from the application program, controls and monitors command execution, and returns operational status information to the application program at command completion. To perform all of these operations, the bubble driver must support the bit/byte level of the bubble memory controller's command and status registers and the "parametric" registers that define the operating mode, system configuration, and extent of the transfer. Depending on the interface level of the application software, the driver itself may be made up of a set of subroutines that are called individually by the host to perform specific bubble system operations.

SOFTWARE-SELECTIVE CONFIGURATIONS

Before you can begin to design a software driver, you must consider all of the selective configurations available within the bubble system and which of these configurations you want to support. As will be explained, the type of data transfer (i.e., direct memory access, interrupt driven, or polled), the data transfer rate, and the selective implementation of the bubble system's error correction feature all are software controlled. The complexity of the driver design depends on the system flexibility desired. For example, a driver may be designed to support only one transfer mode and may not make use of error correction. Conversely, a more generalized driver can be designed to support various transfer modes, data rates, and levels of error correction. The ensuing paragraphs define the driver responsibilities associated with the available software configurations and will aid you in designing a driver that satisfies your specific application.

Data Organization

Probably the most important aspect of the bubble memory system interface is the organization of data. From a software viewpoint, data logically is organized into blocks of bytes called "pages." During data transfer operations, one or more of these pages are transferred between the bubble(s) and the host microprocessor. A page is the smallest increment of data that can be transferred; single bytes cannot be transferred. Conceptually, the data organization within a bubble memory is analogous to a disk system. Just as disk sector sizes are fixed when a disk is formatted, bubble page sizes are established, under software control, when the bubble system is initialized. For a single bubble system, the page size is fixed at either 64 bytes when error correction is implemented or 68 bytes without error correction, and the total number of pages available is 2048. In systems with multiple bubbles, page size can vary from 64 bytes (68 bytes without error correction) to 512 bytes (544 bytes without error correction) depending on the number of bubble devices in the system. Page size is directly proportional to the system data rate and also determines the total number of available pages (address field size). As an example, consider a system consisting of two bubbles (using error correction). With two bubbles, there are two possible ways to configure the system; paralleling the two bubbles for a page size of 128 bytes and a total number of 2048 pages or treating the bubbles serially for a page size of 64 bytes and a total number of 4096 pages. The average data rate for the 128-byte page is 17.0 kbytes per second, and the average data rate for the 64-byte page is 18.5 kbytes per second.

The selection of the appropriate page size depends primarily on the data rate supported by the system. For file system implementation, an additional consideration in page size selection is bubble transfer efficiency versus bubble storage efficiency. Essentially, the following system factors must be weighed:

- Data Rate. The higher the data rate, the faster the microprocessor must respond to the demands of the bubble memory controller. Depending on the data transfer mode selected, some data rates may exceed the data transfer rate of the host microprocessor.
- Bubble Transfer Efficiency. A file consisting of a few large pages can be transferred more efficiently (faster) than a file consisting of a number of small pages.
- Bubble Storage Efficiency. For a typical file system, space must be allocated within each page to link the pages of each file together (individual pages of a file may not necessarily be contiguous). Too large or too small a page size can waste

bubble storage space. If most files are comprised of many small pages (e.g., 64 bytes), a large percentage of bubble storage would be required for establishing the fore/back pointer linkage. Conversely, if most files are smaller than a single page, a large amount of space would remain unused at the end of each page.

Buffering

Buffer operation is an extremely important factor in the reliable transfer of data between the bubble and system memory and is a major consideration in software driver design. A buffer, ideally speaking, is a memory storage area that contains the same amount of data storage as the data block to be transferred. The bubble system's bubble memory controller includes a first-in, first-out (FIFO) 40 byte data buffer that reconciles timing differences between the parallel data transfer to or from the host microprocessor and the serial data transfer to or from the bubble memory. Accordingly, when an application program requests data from a bubble, the software driver is responsible for keeping up with the FIFO for the duration of the data transfer in order to prevent the FIFO from overflowing or underflowing. The specific software driver requirements are dependent on the method of data transfer selected.

Data Transfer Interface Modes

Three distinct software interface techniques can be used to interface host system memory with the bubble system for page data transfer: DMA: interrupt driven, and polled.

In the DMA transfer mode, the BMC operates in conjunction with a DMA controller (e.g., Intel's 8257 or 8237) and uses the DRQ (data request) and DACK/ (data acknowledge) signal lines for establishing the handshake protocol. Assisted by the DMA controller, the BMC transfers data to or from the host system memory. Once the transfer begins, further program intervention is not required until the entire transfer has been completed.

In the interrupt-driven data transfer mode, the DRQ line is connected to an interrupt controller (e.g., Intel's 8259A). During non-DMA data transfers the DRQ line indicates when the BMC's FIFO is half-full (bubble read operations) or half-empty (bubble write operations). This method of interrupting results in a data block transfer arrangement in which the software is responsible for performing the appropriate transfer of data (typically 22 bytes) to or from the FIFO when the interrupt occurs. Using this technique, the software driver only processes the FIFO buffer as needed, and program waits during I/O transfers (polled I/O) are eliminated.

The polled I/O mode is the most simple to implement since no special or external hardware is required to perform data transfers. In the polled I/O mode, the software must determine when to transfer data to or from the FIFO by continually polling a status bit in the BMC's Status Register. This status bit indicates the presence or absence of data in the FIFO on a byte-by-byte basis. The polled I/O mode places significant demand on the host system's processing time since the software continuously must monitor the Status Register to ensure that FIFO overflow or underflow does not occur.

ECC Highlights

The last software controlled option to be considered in the design of your bubble driver is if the built-in error correction circuitry (ECC) within the 7242 Formatter/Sense Amplifier is to be implemented and, if so, what level of error correction is best suited to your application.

Although the inherent data integrity of your bubble memory is extremely high, the incorporation of error correction improves the overall integrity of your system by several orders of magnitude. While boosting the data integrity, the implementation of error correction adds increased overhead to both driver design and host interaction. Since the associated error handling routines can range from simple to complex, you must carefully weigh the software requirements before considering the level of error correction to be supported. In balancing driver and host responsibilities, you must understand the following factors pertaining to ECC:

- The type and nature of errors associated with bubble memories.
- The way in which ECC operates.
- The various levels of error correction available.

All of these factors are explained in detail in a later section.

COMMUNICATING WITH THE BMC

All communications between the host and the bubble memory actually are peformed through the 7220-1 BMC. The BMC has two input/output (I/O) ports; an 8-bit bidirectional data port and an 8-bit command/status port (Figure 1). The port addressed is determined by the least-significant bit of the port address byte. Conceptually, the BMC can be thought of as a disk system controller in that data in the bubble memory is organized into blocks called "pages" that are similar to disk sectors. Information such as starting page location, direction of transfer, and the number of pages to be transferred is passed to the BMC before the desired read or write operation is initiated.

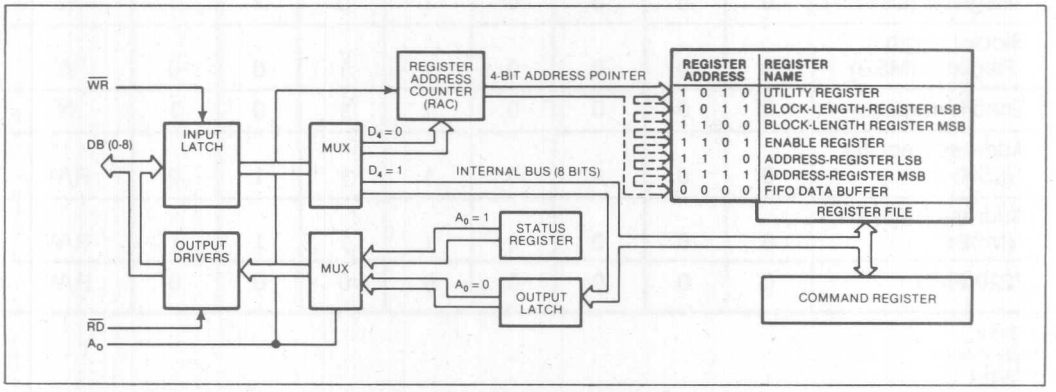


Figure 1. BMC Block Diagram

For simplicity, you can think of the BMC as a 40-byte FIFO (first-in first-out) buffer and a series of six user-accessible 8-bit registers. The FIFO passes data between the outside world and the bubble system's 7242 Formatter/Sense Amplifier and compensates for speed variations. The six registers are loaded prior to most operations and contain information regarding the upcoming transfer and the operating mode of the BMC. Since these registers always are loaded before a command is sent similar to passing parameters to a subroutine before it is invoked, this set of registers is referred to as the "parametric registers." The transfer of data between the host and the BMC's FIFO and parametric registers takes place over the 8-bit data port. The destination (FIFO or parametric register) of the data port transfer is determined by the value in another register called the RAC. As shown in Table 1, bit 4 of the command/status port byte is used to distinguish between a BMC command and either the address of one of the parametric registers or the FIFO.

Table 1. Command Port Function

Function	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Command	0	0	0	1	C	C	C	C
RAC	0	0	0	0	R	R	R	R

When bit 4 is a "one," the low-order four bits are decoded as a BMC command, and when bit 4 is a "zero," the low-order four bits are interpreted as a pointer to the parametric registers/FIFO. This 4-bit register pointer is referred to as the "Register Address Counter" or simply the "RAC."

RAC values that may be written to the command/status port and the registers selected are outlined in Table 2. The RAC points to, or selects, one of the six registers or the FIFO. Once a RAC value is written to the command status port, the next data port read or write operation transfers data between the host interface and the register addressed.

Table 2. Register Address Counter Assignments*

Register Name	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Read/ Write
Utility Register	0	0	0	0	1	0	1	0	R/W
Block Length Register (LSB)	0	0	0	0	1	0	1	1	W
Block Length Register (MSB)	0	0	0	0	1	1	0	0	W
Enable Register	0	0	0	0	1	1	0	1	W
Address Register (LSB)	0	0	0	0	1	1	1	0	R/W
Address Register (MSB)	0	0	0	0	1	1	1	1	R/W
7220 FIFO	0	0	0	0	0	0	0	0	R/W

NOTE(S):
* Write-only bit
* W/A0=1

Referring now to the table, notice that the register addresses are in hexadecimal order from "A" to "F" and that the FIFO has an address of zero. This arrangement of addresses is due to the RAC's auto-incrementing feature. Once a register is selected, each subsequent data port I/O read or write causes the RAC to advance and to point to the next register in the sequence. After the most significant byte of the Address Register is addressed, the RAC advances from F to 0 to point to the FIFO. When it reaches this point it no longer increments. The system now is ready to transfer data into or out of the FIFO without further instructions from the host.

After the FIFO is selected, the RAC stops incrementing and continues to point to the FIFO until the RAC again is accessed through the command/status port. The auto-incrementing feature minimizes the number of instructions required for a given command sequence and ensures that all of the required parametric information is sent to the BMC.

As a user, you are not required to utilize the auto-incrementing feature; each parametric register can be selected and loaded in any order, and specific registers may be updated as required. When individual registers are not accessed in order, each register must be specifically addressed and loaded. Until you become more familiar with the bubble system, the auto-incrementing feature is recommended.

A point to remember is that once a command has been issued to the BMC, the parametric registers must not be updated until the operation is complete. The parametric registers essentially are working registers for the BMC during command execution. When a bubble read or write operation is in progress, the Block Length Register, as explained later in this chapter, contains the terminal page count and is decremented with each page transferred. Attempting to modify this register during command execution would cause the final page count to be incorrect.

The Parametric Registers

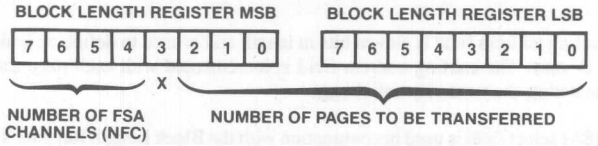
Now that you have been introduced to the Register Address Counter and its operation, let's look at the individual parametric registers addressed by the RAC in more detail.

Utility Register

The Utility Register is a user-defined register that can be both written and read. This register is not incremented or decremented by the BMC. Since the Utility Register is the first register of the RAC sequence, if the register is not used, the least-significant byte of the Block Length Register ihitially can be addressed to eliminate the Utility Register from the sequence. Note that the 4 Mbit bubble memory controller (7224) does not contain a utility register.

Block Length Register

The Block Length Register is made up of two 8-bit registers, a low-order byte register and a high-order byte register. The contents of the block length register determine the system page size and also the number of pages to be transferred in response to a single bubble data read or write command. The Block Length Register or "BLR" is a write-only register that is divided into a terminal count field and a channel field as follows:



The terminal count field is eleven bits in length and is loaded with the total number of pages to be transferred in the ensuing bubble read or write operation. With a field length of eleven bits, from 1 to 2048 pages can be transferred (all zeroes in the field indicates a 2048-page transfer).

The page width (size) is defined by the 4-bit channel field. This field actually specifies the number of formatter/sense amplifier channels available. Note that each 7242 formatter/sense amplifier has two channels to communicate with each bubble memory, therefore the acceptable values in this field select one channel (one half of a bubble memory), two, four, eight, or 16 channels. These field values correspond to page sizes of 32, 64, 128, 256, and 512 bytes (assuming error correction), respectively, when the bubble memories are operated in parallel. (The one-channel mode usually is reserved for diagnostic operations.) Table 3 shows the relationship among page size, channel field value, and formatter/sense amplifier channel selection for parallel bubble operation.

As shown in the table, the channel field bits are encoded and only one bit ever is set in the field.

For example, a channel field value of "0001" selects one bubble memory through channels 0 and 1.

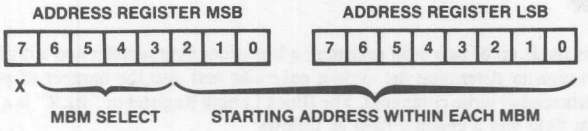
Table 3. FSA Channel Select/MBM Select

MBM Select AP, MSB Bits (6, 5, 4, 3)	"Channel Field" (BLR MSB Bits 7, 6, 5, 4)				
	0000	0001	0010	0100	1000
0 0 0 0	0	0, 1	0, 1, 2, 3	0 to 7	0 to F
0 0 0 1	1	2, 3	4, 5, 6, 7	8 to F	
0 0 1 0	2	4, 5	8, 9, A, B		
0 0 1 1	3	6, 7	C, D, E, F		
0 1 0 0	4	8, 9			
0 1 0 1	5	A, B			
0 1 1 0	6	C, D			
0 1 1 1	7	E, F			
1 0 0 0	8				
1 0 0 1	9				
1 0 1 0	A				
1 0 1 1	B				
1 1 0 0	C				
1 1 0 1	D				
1 1 1 0	E				
1 1 1 1	F				

NOTE(S):
*Normally reserved for diagnostic operations.

Address Register

The Address Register, like the Block Length Register, is made up of two 8-bit registers; a low-order byte register and a high-order byte register. The Address Register is divided into a starting address field and an MBM (Magnetic Bubble Memory) select field show as follows:



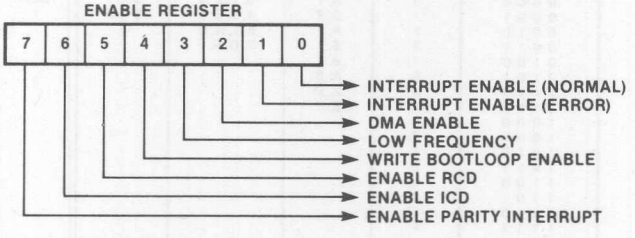
The Address Register's starting address field is eleven bits in length and is used to define on which page of a bubble's 2048 pages that the transfer is to start. The starting address field is incremented with each page transferred during multipage transfers, automatically selecting the next sequential page.

The Address Register's MBM select field is used in conjunction with the Block Length Register's channel field to control the serial selection of bubble memories or groups of bubble memories operated in parallel. To better understand the function of the MBM select field, consider a system consisting of four bubble memories operated as two banks of two bubble memories each.

Referring back to table 3, the channel field in the Block Length Register would be set to "0010" to select two bubbles in parallel and a corresponding page size of 128 bytes. To select between the two banks, the Address Register's MBM select field would be set to "0000" to select the first bank (FSA channels 0 through 3). As page 2048 is transferred to or from the first bank, the Address Register's starting address field rolls over to "0000" and increments the MBM select field to "0001" to select the second bank (FSA channels 4 through 7).

Enable Register

While the Address and Block Length Registers define the system configuration and data transfer, the Enable Register defines the various modes of operation under which the data transfer is performed and defines the conditions under which interrupts can be generated. Several of the Enable Register bits are used individually while other bits are used in combination. Figure 3 shows the individual Enable Register bit definitions.



Interrupt Enable (Normal), when set ("I"), enables the BMC to interrupt the host processor on the successful completion of a command. Conversely, if this bit is not set, an interrupt is not generated on command completion and the host processor must poll the BMC's Status Register to determine when command execution is complete.

Interrupt Enable (Error) is used in conjunction with the Enable RCD and Enable ICD bits to select various error conditions under which the BMC will terminate command execution and interrupt the host processor. The following table outlines the bits combination and corresponding error conditions recognized.

Table 4. Error Correction Options

Enable ICD (bit 6)	Enable RCD (bit 5)	Interrupt Enable (Error) (bit 1)	Interrupt Condition
0	0	0	No interrupt on error
0	0	1	Interrupt only on timing error
0	1	0	Interrupt on uncorrectable or timing error
0	1	1	*Interrupt on uncorrectable, correctable or timing error
1	0	0	Interrupt on uncorrectable or timing error
1	0	1	Interrupt on uncorrectable, correctable or timing error
1	1	0	Illegal
1	1	1	Illegal

NOTE(S):
*Normally not used.

DMA Enable, when set, enables the BMC to operate in the DMA data transfer mode. In the DMA mode, a DMA controller is interfaced to the BMC and DRQ-DACK/protocol is used to perform byte transfers. Note that in the DMA mode, the BMC activates its DRQ output *each time* it places a byte in the FIFO (bubble read operation) or each time there is room for *at least one byte* in the FIFO (bubble write operation). When the DMA Enable bit is not set, the BMC operates in the interrupt driven or polled mode. In either of these modes, the BMC's DRQ output goes active when 22 or more bytes are present in the FIFO during a bubble read operation or when there is space for 22 bytes during a bubble write operation. Note that if the DRQ is not used (i.e., polled mode), the host processor must examine the Status Register's FIFO Ready bit to determine when data should be taken from or written to the FIFO.

MFBTR, (Maximum FSA to BMC Transfer Rate), determines the maximum burst transfer rate from the FSA(s) to the BMC FIFO. This bit only applies to bubble read operations and is effective only during single-page transfers or during the transfer of the last page of a multipage transfer. Table 3 shows the effect of MFTBR bit on the transfer rate for parallel bubble operation.

Write Bootloop Enable, when set, enables the bootloop to be rewritten. Conversely, if this bit is not set and a Write Bootloop command is received, the command is aborted immediately and the Timing Error is set in the Status Register. Since writing the bootloop only is performed as a diagnostic test or to recover from a system failure, this bit provides protection against accidental rewrite of the bootloop data.

Enable RCD (Read Corrected Data), when set, causes the BMC to issue a Read Corrected Data instruction to all the FSAs in the system in response to one or more FSAs reporting an error. On receipt of the instruction, each FSA cycles the erroneous page through its error correction logic and then transfers the page to the BMC. For any FSA not reporting an error, the data harmlessly cycles through the error correction logic. When the page transfer is complete, the BMC interrogates the FSA to determine if the error was correctable (if the error was uncorrectable, erroneous data would have been transferred to the BMC).

Enable ICD (Internally Correct Data), when set, causes the BMC to issue an Internally Correct Data instruction. Any FSA reporting an error causes the instruction to be issued to all FSAs in the system.

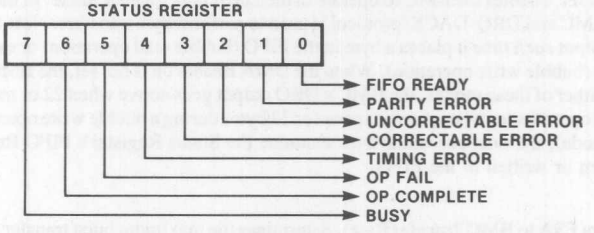
On receipt of the instruction, each FSA cycles the data through its error correction logic (regardless of whether it contained an error), but does not transfer the page to the BMC. After cycling the page, each FSA reports its error Status (correctable or uncorrectable) to the BMC. The FSA not reporting an error harmlessly cycles through the error correction logic and reports a correctable error to the BMC. Note that both the Enable RCD and Enable ICD bits cannot be set at the same time.

Enable Parity Interrupt, when set, enables the BMC to interrupt the host processor when it detects a parity error on a data byte sent from the host processor (the BMC automatically checks for odd parity on each data byte received from the host processor and implements odd parity on each byte sent to the host processor). When the Enable Parity Interrupt bit is not set and parity error is detected, no interrupt is generated (following the transfer, the Parity Error bit in the Status Register will be set).

Status Register

As stated at the beginning of this chapter, the host processor executes an I/O read instruction with the BMC's A0 address input equal to "1" to access the Status Register. As will become evident in the individual Status Register bit descriptions, the Status Register is read in response to interrupts from the BMC and, during polled data transfers, is read continually to determine when data is to be written to or read from the BMC's FIFO.

The Status Register also provides information regarding error conditions, the completion or termination of commands, and the BMC's readiness to accept new commands. The individual Status Register bits are shown as follows.



Note that bits 1 through 6 are valid only when the Busy bit (bit 7) is not set and that these bits are cleared whenever a new command is received. The Status Register also can be cleared by writing to the register address counter (RAC) with bit 5 set to "1" and any valid parametric register address (0AH through 00H).

Busy. Bit 7, the Busy bit, is set ("I") when the BMC is in the process of executing a command. The BMC sets its Busy bit shortly after a command is received (the Busy bit must be clear in order for the BMC to accept any command other than an Abort command) and keeps Busy set until the operation is complete (or until an operation is halted because an error has occurred.)

It is important to note that the Busy bit remains set until all other status bits have been updated and that it therefore is possible to see illogical bit combinations such as Busy and Op Complete at the same time. With the exception of the FIFO Available bit, all other Status Register bits should be considered valid only after the Busy bit returns to an inactive ("0") level.

OP Complete, Bit 6, Op Complete, is set when the BMC successfully completes the execution of a command.

OP Fail, Bit 5, Op Fail, is set when the BMC is unable to complete execution of a command. In general, this bit is valid only after the Busy bit returns to an inactive level; other error bits in the Status Register will be set to indicate why the operation failed.

Timing Error, Bit 4, Timing Error, is set for several error conditions. The most frequent cause of timing errors is when the host processor cannot keep up with the rate at which the BMC fills or empties its FIFO (referred to as an overflow or underflow condition). Timing errors also occur if the correct number of bits is not set in an FSA's bootloop register (to function properly, an FSA must have either 272 loops active when error correction is not implemented or 270 loops active when error correction is implemented). This condition occurs during a Read command if a mistake is made either when the bubble's bootloop is written or if the bootloop register is loaded incorrectly from the user's system. Another source of timing error is if no bootloop sync code is found during an Initialize or Read Bootloop command. The last source of a timing error is when a Write Bootloop command is received by the BMC and the Write Bootloop Enable bit is not set in the Enable Register. Of the preceding sources, regular or periodic occurrences of timing errors usually indicate an inherent inability of the host processor to meet the bubble's data transfer requirements and will be most apparent during bubble read operations, especially if the MFBTR bit is set (i.e., MFBTR=0 in the Enable Register).

Correctable Error, Bit 3, Correctable Error, is set when an FSA reports that a correctable error was detected during the last bubble read operation. Depending on the level of error correction selected, the setting of this bit indicates a correctable error in the current page held by the FSA, in the last page read, or in one of the pages read during a multipage transfer.

Uncorrectable Error, Bit 2, Uncorrectable Error, is set when an FSA reports an uncorrectable error during the last bubble read operation. Like the Correctable Error bit, the setting of this bit depends on the level of error correction selected and indicates an uncorrectable error in either the last page transferred or in the page currently held by the FSA.

Parity Error, Bit 1, Parity Error, is set when the BMC detects a parity error on the data byte sent from the host during a bubble write operation. If the Enable Parity Interrupt bit is set in the Enable Register, the BMC will terminate the write operation and interrupt the host processor when the parity error is detected.

FIFO Available, Bit 0, FIFO Available, is unique in that it is the only bit the Status Register that is valid when the Busy bit is set. During data transfer operations (i.e., when the Busy bit is set), the FIFO Available bit acts as a gate for the host microprocessor's data handling software. During bubble write operations, if the FIFO Available bit is set there is room for more data in the FIFO and the host microprocessor can proceed with the transfer. Conversely, if the FIFO Available bit is not set, the FIFO is full and the host must wait for the BMC to "catch up" before sending more data. During bubble read operations, if the FIFO Available bit is set, data has been placed in the FIFO by the BMC and can be taken by the host microprocessor; if the FIFO Available bit is not set, the data is not yet available.

FIFO

The BMC's first-in first-out (FIFO) buffer passes data between the user interface and the FSA. The primary purpose for the FIFO is to reconcile timing differences between both the user interface and the BMC and between the BMC and the FSAs.

The FIFO itself is 40 bytes wide and, including the FIFO's input and output latches and the BMC's input latch, can store up to 43 bytes. The FIFO is "dual ported" in that data can be written into one port while simultaneously being read from the other port.

When the BMC is busy executing a command, the FIFO functions as a data buffer, and when the BMC is not busy, the FIFO is available for use as a general-purpose FIFO. For this reason, the BMC is said to be in the general-purpose FIFO mode when it is not busy. During execution of commands that involve the transfer of data between the user interface and the FSAs, the data passes through the FIFO, and the status of the FIFO is indicated by the FIFO Ready bit in the BMC's Status Register. When the FIFO Ready bit is set ("one") during a write operation, there is space in the FIFO for more data, and when this bit is set during a read operation, data is present in the FIFO.

The BMC's DRQ output also indicates FIFO status. In the DMA data transfer mode, DRQ is used in conjunction with the DACK/ input from a DMA controller (e.g., in Intel 8257 or 8237) on the user interface to provide a standard DMA data transfer protocol. In the non-DMA transfer mode (polled or interrupt-driven data transfers), the DRQ output indicates that the FIFO is either half full or half empty according to the direction of the data transfer; during a write operation, DRQ is set when there is space for 22 more bytes, and during read operation, DRQ is set when there are 22 bytes present in the FIFO. Accordingly, when performing non-DMA data transfers, blocks of 22 bytes should be transferred to or from the FIFO so that the host processor does not spend a disproportionate amount of time servicing the DRQ-initiated interrupt or polling the BMC's Status Register.

The FIFO automatically is addressed after the last parametric register is written due to the self-incrementing nature of the Register Address Counter. Alternatively, the FIFO can be addressed explicitly by writing to address 0 of the Register Address Counter. Note that since byte 0 of the FIFO is cleared whenever the Register Address Counter is addressed, writing the parametric registers (or to the FIFO itself) must be avoided while the FIFO contains valid data. Also, when using the FIFO in the general-purpose mode, the host system is responsible for resetting the FIFO prior to any data transfer.

During read and write operations, the host system is responsible for keeping up with the data transfer in order to prevent a FIFO overflow or underflow condition. If the FIFO overflows or underflows during the data transfer, the operation is aborted and the Op Fail and Timing Error bits are set in the BMC's Status Register.

The FIFO AVAILABLE bit is also set whenever the RAC is not pointing to the BMC FIFO (RAC address = 00H). When writing the parametric registers, for example, if the user reads the status between say the Enable Register and the Address Register, the status byte would indicate FIFO AVAILABLE. This status value is forced by internal BMC logic.

COMMANDS

The BMC's command set consists of 16 commands that are selected by a 4-bit command code. As previously described, commands are passed to the BMC by writing to the BMC's command port with bit 4 of the command byte set to "1" (the Register Address Counter is addressed when bit 4 is "0"). Table 5 lists the 4-bit command codes and the corresponding commands.

For most commands sent to the BMC, the parametric registers must be loaded with operating information before the command is sent. Also, with the exception of the Abort command, commands cannot be issued to the BMC while another command is being executed (i.e., when the Busy bit in the Status Register is set). The remainder of this section offers brief descriptions of the BMC's command set; descriptions of command usage are presented in succeeding sections. Table 6 of this section presents a summary of command execution times.

The 16 commands can be grouped into categories according to their frequency of use. The most commonly used commands are:

- Abort
- Initialize
- Read Bubble Data
- Write Bubble Data

Other commands used during normal bubble system operation include:

- Read Seek
- Write Seek
- MBM Purge
- Read Corrected Data
- Reset FIFO
- Software Reset
- Read FSA Status
- Read Bootloop

The remaining commands are related to bootloop operation and are used only for diagnostic purposes:

- Read Bootloop Register
- Write Bootloop Register
- Write Bootloop Register Masked
- Write Bootloop

Abort

The Abort command (unlike any other command), when issued while the BMC is busy executing another command, causes the termination of the command being executed. On receipt of this command, the MBMs are stopped in an orderly manner to prevent the loss of bubble data. Since the Abort command is the only command recognized by the BMC while it is busy, this command is issued following power-up or whenever the BMC is in an unknown state. The Abort command requires no prior loading of the parametric registers.

When an Abort command is issued while the BMC is not busy, the command functions as an FIFO Reset to clear any data present in the FIFO. An Abort command issued when the MBM drive coils are active (i.e., data transfer command is executing indicated by the Busy bit in the Status Register) must be followed by an Initialize command.

Table 5. BMC Command Set

D3	D2	D2	D1	Command Name
0	0	0	0	Write Bootloop Register Masked
0	0	0	1	Initialize
0	0	1	0	Read Bubble Data
0	0	1	1	Write Bubble Data
0	1	0	0	Read Seek
0	1	0	1	Read Bootloop Register
0	1	1	0	Write Bootloop Register
0	1	1	1	Write Bootloop
1	0	0	0	Read FSA Status
1	0	0	1	Abort
1	0	1	0	Write Seek
1	0	1	1	Read Bootloop
1	1	0	0	Read Corrected Data
1	1	0	1	Reset FIFO
1	1	1	0	MBM Purge
1	1	1	1	Software Reset

Table 6. 7220-1 Command Execution Times

Four-Bit Command Code (Hex)	Command	Description	Performance
0	LRBLRMSK	1 FSA channel selected 2 FSA channel selected	900 μs 900 μs
1	Initialize	Best case (N = #MBM) Worst case	350+(85,200) N μs 350+(164,740) N μs
2	Read	Single page (MFBTR=0) Single page (MFBTR=1) N page transfer (MFBTR=0) N page transfer (MFBTR=1)	t_{SEEK}+8690 μs t_{SEEK}+12,770 μs t_{SEEK}+8690+(7500) (N−1) μs t_{SEEK}+12,770+(7500) (N−1) μs
3	Write	1 page transfer N page transfer	t_{SEEK}+7450 μs t_{SEEK}+7450+(7500) (N−1) μs
4	Read Seek	Best case Worst case	7350 μs 89,250 μs
5	Read BLR	Any number of FSA channels	900 μs
6	Write BLR	Any number of FSA channels	900 μs
7	Write BL	Single bubble selected	82,850 μs
8	Read FSA Status	1 bubble in system N bubbles in system	75 μs 75+40 (N−1) μs
9	Abort	1 bubble in system N bubbles in system	100 μs 100+40 (N−1) μs
A	Write Seek	Best case Worst case	7350 μs 89,250 μs
B	Read BL	Best case Worst case	86,000 μs 165,000 μs
C	Read Corrected Data	Any number of FSAs selected	1400 μs
D	FIFO Reset	N/A	50 μs
E	MBM Purge	N/A	150 μs
F	Software Reset	N/A	50 μs

Initialize

The Initialize command prepares the bubble system for subsequent operations and is used when the bubble system is powered up (following the Abort command) or whenever the system's error correction implementation is changed. The Initialize command effectively performs the following BMC commands:

- Abort
- MBM Purge
- FIFO Reset
- Read Bootloop
- Write Bootloop Register Masked

When an Initialize command is received, all internal registers within the BMC are cleared and the FIFO is reset. The BMC then reads the bootloop from each bubble and writes the corresponding bootloop information into the bootloop registers of each FSA. The bubble is left positioned at page zero (logically) corresponding to the values set in the BMC page address counters. Before an Initialize command can be issued, the following information must be loaded into the parametric registers:

- The channel field in the Block Length Register must be set to "0001" to arrange all bubbles in the system in a serial configuration to allow the individual bootloops to be read from each bubble and subsequently written to the bootloop registers of the corresponding FSA channels.
- The MBM select field in the Address Register must select the last (highest numbered) bubble in the system to inform the BMC of the number of bubbles present within the system (for a one bubble system the value would be "0000").
- The bits selecting error correction in the Enable Register must be set according to how subsequent read and write operations are to be performed (with or without error correction) since the number of 1's written to the FSA bootloop registers is not the same with error correction implemented as when error correction is not implemented. Note that simply switching between ECC modes (level 1, 2, or 3) does not require re-initialization.

Read Bubble Data

The Read Bubble Data command causes data to be read from the MBMs and into the BMC's FIFO. Immediately before the Read Bubble Data command is issued, the host computer must load, with the exception of the Utility Register, all of the parametric registers. Specifically, the following parametric information must be loaded prior to command execution:

- The channel and "number of pages to be transferred" in the Block Length Register must be set to define the page size (number of FSA channels) and number of pages to be transferred.
- The appropriate bits must be set in the Enable Register to select the transfer mode (DMA or non-DMA) and interrupt sources; if error correction is to be used (i.e., if the FSA bootloop registers have been initialized for error correction), the level of error correction must be selected.
- The MBM select and starting address fields in the Address Register must be set to define the (first) MBM and page within the MBM where the transfer is to occur.

Write Bubble Data

The Write Bubble Data command causes data read from the BMC's FIFO to be written into the MBMs. Immediately prior to issuing the Write Bubble Data command, the host computer must load the Block Length, Enable, and Address Registers as described for the Read Bubble Data command. Data should not be loaded into the BMC FIFO until after the command has been issued. Note that a write data transfer does not begin until at least two bytes of data have been loaded into the BMC FIFO.

Read Seek

The Read Seek command is used to reduce system access time of a subsequent Read Bubble Data command by positioning the page to be read at the selected bubble's output track. Note that although the Read Seek command does not cause any data to be transferred, the following information must be loaded into the parametric registers before the command is issued:

- The channel field in the Block Length Register must specify the page size (number of FSA channels).
- The error correction bits in the Enable Register must be set identical to the values used when the system was initialized.
- The MBM select field in the Address Register must be set to select the bubble containing the page to be read and the address field must specify a page number that is one less than the (first) page to be read by the subsequent Read Bubble Data command.

Write Seek

The Write Seek command is used to reduce the system access time for a subsequent Write Bubble command by positioning the page to be written at the selected bubble's input track. Note that like the Read Seek command, the Write Seek command does not cause any data to be transferred. Similarly, the following information is issued:

- The channel field in the Block Length Register must specify the page size (number of FSA channels).
- The error correction bits in the Enable Register must be set identical to the values when the system was initialized.
- The MBM select field in the Address Register must be set to select the bubble containing the page to be written and the address field must specify a page number that is one less than the (first) page to be written by the subsequent Write Bubble Data command.

MBM Purge

The MBM Purge command is used in place of the Initialize command when the bootloop register is to be loaded from an external source (once a bootloop has been identified, the bootloop register pattern can be maintained by the host and loaded directly from external memory with a Write Bootloop Register or Write Bootloop Register Masked command to conserve power and increase speed during an initialization sequence). The MBM Purge command clears the BMC's internal registers and the address field of the Address Register. The MBM select field in the Address Register and the other parametric registers are not cleared. Like the Abort command, the MBM Purge command does not require the loading of the parametric registers prior to command execution.

Read Corrected Data

The Read Corrected Data command is used only when error correction is implemented and only is applicable when error correction level 2 or 3 is selected (the Read Corrected Data command is issued automatically by the BMC when error correction level 1 is selected). When an FSA reports a correctable error, the Read Corrected Data command is issued to cause the corrected page in the FSA(s) to be read into the BMC's FIFO. Note that since the parametric registers previously were loaded for the read operation in which the error was detected and since the address field is not incremented (i.e., the address of the page in error is in the address field), it is not necessary to load the parametric registers prior to issuing the Read Corrected Data command.

Reset FIFO

The Reset FIFO command causes the BMC's FIFO (and its input and output latches) to be cleared. Normally, this command is issued prior to issuing the Write Bootloop, Write Bootloop Register, or Write Bootloop Register Masked commands to ensure that the FIFO will accept 40 bytes; the Reset FIFO command does not require loading of the parametric registers.

Software Reset

The Software Reset command clears the BMC's internal registers, and the terminal count field in the Block Length Register and starting address field in the Address Register. Additionally, the Software Reset command causes the BMC to clear the status register of every FSA channel in the system. Since the Software Reset command does not clear the FSA channel and MBM select fields in the Block Length and Address Registers or any of the Enable Register bits, it is not necessary to reinitialize the parametric registers following a Software Reset command, and no loading of the parametric registers is required prior to command execution.

Read FSA Status

The Read FSA Status command causes the BMC to read the 8-bit status register of each FSA channel in the system (the number of FSA channels specified in the channel field of the Block Length Register). This command is issued to determine the number of FSAs in a system. Following command execution, the individual status register bytes will be in the BMC's FIFO in ascending order; one byte per FSA channel. All values returned to the host will be 20H or 28H if error correction is enabled. This occurs because the FSA status is cleared when read. The BMC always reads the FSA status prior to interrupting (or informing) the host of an error. Therefore, the Read FSA status command can only read the "cleared" FSA status values (20H or 28H). No loading of the parametric registers is necessary prior to command execution.

Read Bootloop Register

The Read Bootloop Register command causes the BMC to read the bootloop register of the selected FSA channel into its FIFO. This command is used initially to ensure that the bubble system is communicating properly (bubble-to-FSA and FSA-to-BMC communication established) and is used to transfer bootloop information to the host system for subsequent bootloop initialization from an external source (e.g., user EPROM). Prior to command execution, the channel field in the Block Length Register and the MBM select field in the Address Register must be loaded with the number of FSA channels (normally two) and the corresponding bubble (FSA channel pair) to be selected. Note that since each individual FSA channel's bootloop register contains 20 bytes, reading a pair of FSA channels fills the BMC's 40-byte FIFO; reading the bootloop registers of more than two channels is possible but not recommended since data must be taken from the FIFO to avoid an overflow condition. Also, since the bootloop register data from each FSA channel pair actually is interleaved on a bit-by-bit basis before it is assembled into bytes, reading the bootloop register for a single channel likewise is not recommended.

Write Bootloop Register

The Write Bootloop Register command causes the BMC to write the contents of its FIFO into the bootloop register of the selected FSA channel(s). This command (and the Write Bootloop Register Masked command) is used during bubble system initialization when the bootloop is written from an external source rather than from the bubble itself. In order to use the Write Bootloop Register command, the bootloop register data must be loaded into the FIFO prior to command execution and the channel and MBM select fields of the Block Length and Address Registers must be loaded with the number of FSA channels (normally two) and the corresponding bubble (FSA channel pair) to be selected. Recalling that each individual FSA channel's bootloop register contains 20 bytes, 40 bytes normally are written into the FIFO to initialize the two FSA bootloop registers associated with each bubble. However, a 41st byte of zeroes *must* be written to ensure a successful command execution status. Note that the parametric registers should be loaded prior to loading the FIFO.

Proper operation of the bubble system requires that each FSA bootloop register contains either 135 (error correction selected) or 136 (error correction disabled) logic "1" bits that correspond to the 135 or 136 valid data storage loops.

Write Bootloop Register Masked

The Write Bootloop Register Masked command is identical to the Write Bootloop Register command previously described with the exception that the number of "1" bits written automatically is stopped when the required 135 or 136 logic "1" bits have been written for each channel. The parametric registers are loaded as described for the Write Bootloop Register command; the number of logic "1" bits written (135 or 136) is determined by the setting of the error correction bits in the Enable Register.

Write Bootloop

The Write Bootloop command causes the existing contents of the selected bubble's bootloop to be replaced by the 40 bytes of information currently contained in the BMC's FIFO. This command is used only after it has been determined that the existing bootloop is invalid (i.e., a storage loop previously identified as "good" has become defective) and typically is not required for the life of the bubble system. Remember that the parametric registers must be loaded prior to pre-loading the FIFO with the 40 bytes of information to be written followed by a 41st byte of zeroes.

If it should become necessary to use the Write Bootloop command, the channel field in the Block Length Register and the MBM select field in the Address Register must be loaded with the number of FSA channels and the corresponding bubble to be selected. As an additional safeguard, the Write Bootloop Enable bit in the Enable Register additionally must be set (if this bit is not set, command execution immediately will be aborted and the Timing Error bit will be set in the Status Register).

ERROR CORRECTION

As mentioned earlier in this chapter, several factors pertaining to error correction must be understood and weighed according to your specific application before implementing error correction. From a software perspective, the selection of the appropriate level of error correction is based on the host system's requirements relative to error logging and data recovery. Before describing the individual levels of error correction and the associated software requirements, the types of errors that can occur within bubble memories are described as a preface to error correction.

Bubble Errors

To understand the function and implementation of error correction, the differences between bubble errors and semiconductor device errors must be distinguished. In conventional semiconductor memory, errors are classified as either hard (non-recoverable) or soft (recoverable). Usually, hard errors are the result of physical or irreversible damage within the device itself (e.g., oxide breakdown or junction burnout), and soft errors are the result of transient conditions and generally do not reappear when the data is rewritten.

Unlike semiconductor devices, bubble memory devices have no active elements and, as such, rarely experience hard (non-recoverable) errors. Accordingly, all bubble memory errors can be considered as soft errors (for information on irreversible failure mechanisms in bubbles, refer to the Intel 7110 Bubble Memory Reliability Report, RR-36 Order Number 210632). In order to further define the nature of soft errors in bubbles, errors are classified either as "data" errors or "read" errors. A data error occurs when a data inversion occurs within a storage loop; the data is lost, but since no physical damage has occurred, the data can be rewritten and the bubble can remain operational. Data errors typically occur when the bubble is operated beyond its safe operating region and, as such, are seldom encountered during normal operation. The most common type of soft error is a "read" error that occurs during a bubble read operation, usually as a result of noise in the detection/sense circuitry. Since the data in the storage loop is unaltered during a bubble read, the data can be recovered by simply repeating the read operation.

Error Detection/Correction Capability

In respect to the Formatter/Sense Amplifier (FSA), errors either are correctable (the FSA is able to reconstruct the data using an error correction algorithm before the data is transferred to the BMC) or uncorrectable, irrespective of the type of error (data error or read error). The error correction code used by the FSA is a 14-bit Fire code that is appended to each 256-bit block of data. This code is capable of correcting all single error bursts up to, and including, five bits in length and has proven to be well suited to the error model for the 7110 MBM. Table 7 outlines the FSA's error correction capability and probability of page errors; the bit error rate can be obtained by dividing the "probability of page in error" by the number of bits per page.

ECC Options

The FSA's error correction circuitry (ECC), in conjunction with the BMC, provides three levels of error correction. Each level places unique demands on the host system that range from simple data recovery to the logging of specific pages in which the error occurs. The desired level of error correction is selected by the setting of the appropriate bit or bits within the BMC's Enable Register. Table 8 defines the relevant Enable Register bits for the three levels of error correction available.

Table 7. FSA Error Detection/Correction Capability

Type of Error	Approximate Probability of Page in Error	% Correction	% Detection
Single Read Error	10^{-6}	100	—
Single Data Error	10^{-7}	100	—
Random Double Read Error	10^{-12}	—	100
Read Error Burst Length 2	—	100	—
Data Error Burst Length 2	—	100	—
Read Error Burst Length 3/4	10^{-9}	100	—
Random Double Data Error	10^{-14}	—	100
Read Error Burst Length 5	—	100	—
Random Triple Read Error	10^{-17}	—	100
Single Soft + Read Burst 2	—	—	100
Undetected/Uncorrected Error Escape Rate	10^{-13}	—	—

NOTES:

1. Read errors are recoverable by retry or error correction.
2. Data errors are recoverable by error correction methods only.

Table 8. Error Correction Level Selection

Error Correction Level	Enable Register Bit		
	Bit 6 ICD	Bit 5 RCD	Bit 1 Interrupt Enable (Error)
Level 1	0	1	0
Level 2	1	0	0
Level 3	1	0	1

In typical bubble memory systems, the prevention of the erroneous transfer of data to the host is primary concern, and the actual location of the error (error logging by page) is secondary. This fact is especially true if the error is correctable. Both the prevention of transferring erroneous data and error logging of the page in error, however, can be satisfied by error correction, and you must select the error correction level that best fits your specific application.

ECC Operation

When error correction is implemented, each page of data written to the MBM contains additional bits for the ECC code. For example, in a single-bubble system like the BPK 72, the page size decreases from 68 bytes to 64 bytes per page when error correction is implemented. As each page of data is read from the MBM and assembled into the FSA, the FSA's error correction circuitry checks the integrity of the data. When an error is detected (in a completely assembled page), the FSA notifies the BMC (by activating its ERR.FLG/ line) and sets the appropriate bits within its internal status register according to whether or not the error is correctable. The BMC, depending on the level of error correction selected and the nature of the error (correctable or uncorrectable), either issues a command to the FSA to continue the transfer (transparent to the host) or interrupts the host and waits for additional instructions before proceeding.

The point at which the data is transferred between the FSA and the BMC's FIFO is the key to understanding the difference among the three levels of error correction. Once this timing is understood, it will become easier to see how the levels differ in terms of the interrupts generated and the intervention required by the host (software driver) during the transfer. Note that while the BMC generates an interrupt when an error is detected, it is not mandatory for the host to support interrupts; the host can read the BMC's Status Register at the completion of command execution to determine the nature of the error condition.

During a bubble read operation with error correction enabled, the FSA senses and formats the data and places the data in its two 270-bit serial FIFOs (one FIFO for each channel). The data is read in the fully buffered mode where a full page (512 bits) plus the additional 28 ECC bits are read into the two FIFOs before any data is transferred to the BMC's FIFO. By fully buffering the data, the FSA can detect an error and notify the BMC before any data is transferred. If no error is detected by the FSA, the contents of the FSA's FIFOs are written to the BMC's FIFO while the next page from the MBM is being read into the FSA (note that the 28-bits of ECC code are never transferred to the BMC's FIFO). In order to correct an error once it has been detected, the FSA cycles the data through its error correction network. Once the data is cycled, the "corrected" data is either automatically transferred to the BMC (transparent to the host) or the data is held in the FSA FIFOs awaiting further instruction from the host. As an alternative to using the "corrected" data (error correction level dependent), the page in error could simply be reread when an error was reported since a majority of the errors encountered are "read" errors. The following paragraphs describe ECC operation during each of the three levels of error correction.

Level 1

Level 1 is the minimum level of error correction and is used only when the host system is concerned with maintaining bubble data integrity. As an example of this type of application, consider a bubble-based operating system that is downloaded into user RAM for execution. With this application, the primary concern is that the entire operating system is transferred correctly rather than where the error occurred. As will be seen, Level 1 is well suited to applications where go/no-go types of data transfers are required.

If an error is detected during Level 1 operation, the FSA activates its ERR.FLG/ line to the BMC. The BMC, in response, automatically issues a Read Corrected Data (RCD) command to the FSA which causes the FSA to cycle the data through its ECC network and update its status register, and then to transfer the data to the BMC. If the soft error was correctable, valid data would have been transferred to the BMC; the BMC would increment its Address Register to the next page to allow the operation to continue, and an interrupt would not be generated (i.e., the entire operation would be transparent to the host system). However, if the soft error was beyond the capability of the FSA's error correction circuitry (i.e., an uncorrectable error), invalid data would have been transferred to the BMC. The BMC, after reading the "uncorrectable error" status from the FSA, stops command execution and interrupts the host, and does not increment its Address Register. Since the host is aware that an uncorrectable error has occurred, the proper response is to repeat the entire read operation (by reloading the parametric register and reissuing the read command) since the erroneous page already has been transferred to the host.

Note that since the BMC does not increment its Address Register after reading the "uncorrectable error" status from the FSA, it is possible to perform page-specific logging of uncorrectable errors.

Level 2

Level 2 differs from Level 1 in that the transfer of erroneous (uncorrectable) data to the BMC is prevented and successive retries of the page in error are possible since the Address Register is not incremented. As with Level 1, correctable errors are transparent to the host system.

When a soft error is detected during Level 2 operation, the BMC automatically issues an Internally Corrected Data (ICD) command to the FSA.

In response to this command, the FSA cycles the data through its error correction network and updates its status register to indicate if the error is correctable or uncorrectable, but does not transfer the data to the BMC. The BMC, in turn, reads the FSA's status register and updates its own Status Register. If the error is correctable, the BMC automatically issues an RCD command to the FSA (to transfer the corrected data) and increments its Address Register to the next page address to allow the read operation to continue. Like Level 1 operation, the transfer of the corrected page is transparent to the host; the subtle difference between Level 1 and Level 2 is the time required to execute since the data is cycled through the error correction network twice (first by the ICD command and again by the RCD command). While the second correction cycle does not change the data, it does, however add approximately 350 milliseconds to the transfer operation.

If the soft error is uncorrectable, command execution is halted on the page in error and the BMC interrupts the host system. When interrupted, the host system can read the BMC's Address Register to determine the page in error and can issue a subsequent Read command (without reloading the parametric register) to retry the page. If, when the page is reread, the error does not recur (i.e., if the uncorrectable error is a "read" error), command execution continues with the next page. If successive retrys are unsuccessful (i.e., if the uncorrectable error is a "data" error), the page most likely will have to be rewritten. The host system, however, can examine the erroneous page by issuing an RCD command to the BMC to transfer the uncorrectable data from the FSA. Following the transfer of the erroneous page, the BMC again will interrupt the host and will not increment its Address Register. Note that the uncorrectable data transferred to the BMC will not necessarily match the erroneous data originally read from the MBM since the FSA's error correction circuitry attempts to correct the data.

Level 3

Level 3 offers the most complete means of error handling and, at the same time, is the most software intensive level since the host system is interrupted when either a correctable or uncorrectable error is encountered. Accordingly, error logging may be performed on pages containing correctable as well as uncorrectable errors, and an unlimited number of retrys can be attempted on the erroneous page. As with Level 2, the transfer of erroneous data to the BMC can be prevented.

When a soft error is detected during Level 3 operation, the BMC automatically issues an ICD command to the FSA. Like Level 2 operation, the FSA cycles the data through its error correction network and updates its status register, but does not transfer the data to the BMC. The BMC, in turn, reads the FSA status register, updates its own Status Register, and interrupts the host system even if the error is correctable. When interrupted, the host system must examine the BMC's Status Register to determine if the error is correctable or uncorrectable and can log the address of the page in error by reading the BMC's address register (the Address Register is not incremented). Note that it is not necessary for the host to support interrupts as the host can continuously poll the BMC's Status Register for an error interrupt.

If the error is correctable, the host system can either issue an RCD command (to transfer the corrected data from the FSA to the BMC) or can retry the page by issuing subsequent Read commands. Note that by retrying pages with correctable errors, the host system can distinguish between "read" and "data" errors since consistently correctable errors on a page indicate a "data" error (a "read" error would not recur when the page was reread).

If the error is uncorrectable, the host system only would retry the erroneous page by issuing additional Read commands and, if successive retrys were unsuccessful, would have to rewrite the page. As with Level 2 operation, the host system can issue an RCD command to transfer the uncorrectable data from the FSA.

Status Register

When using error correction, the host system can read the BMC's Status Register at the time of the interrupt or at the completion of command execution to ascertain additional information relating to the error condition. The relevant bits of the STR are bits 6, 5, 3 and 2; the remaining bits are irrelevant to error correction.

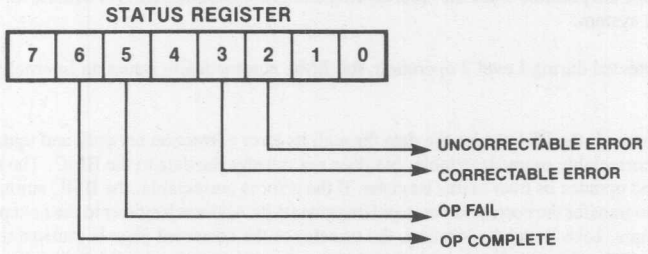


Table 9 lists the possible status indications that could occur when using error correction.

Table 9. Error Correction Status Indications

STR Bit Set	Error Correction Level	Indication
3	3	Correctable error detected, address of page containing the error is in the AR.
5 and 2	All	Uncorrectable error detected, address of page containing the error is in the AR.
6	All	Operation complete, no errors detected.
6 and 3	1 and 2	Operation complete, one or more correctable errors detected and corrected.
6 and 2	1	Operation complete, uncorrectable error detected on last page of multipage transfer.
5, 3 and 2*	1 and 2**	Very Rare. During a multipage transfer, one or more correctable errors detected and corrected on one or more pages, and a subsequent uncorrectable error detected on a page other than the last page of the specified transfer.
6, 3 and 2*	1	Extremely rare. During a multipage transfer, one or more correctable errors detected and corrected on one or more pages, and a subsequent uncorrectable error detected on the last page of the specified transfer.

*These status indications may occur on single-page transfers in Multi-MBM systems when one MBM has a correctable error and another MBM has an uncorrectable error.

**This status indication may occur in level 3 in Multi-MBM systems when more than one MBM has an error on the same page and at least one of the errors is correctable and one of the errors is uncorrectable.

DRIVER DESIGN

This section contains specific software interface examples that outline important considerations associated with the various ways in which a bubble system can be operated. As will be explained, command execution can be performed either in an interrupt driven mode or in a polled mode irrespective of the data transfer mode (polled, interrupt-driven, or DMA) selected to effectively provide the six unique operating modes shown in Figure 2. Since both polled and interrupt driven command execution are common to all three data transfer modes, details concerning command execution are discussed prior to examining specific examples of each data transfer mode. Before you begin to design your software driver, it is recommended that you thoroughly understand the information presented in this section.

Command Execution

To better understand the software driver's responsibilities, it is helpful to separate command execution into two phases; a command phase and a result phase. During the command phase, the driver generally (command dependent) loads the parametric registers, issues the desired command, then verifies that the command has been accepted; during the result phase, the driver determines the success or failure of the command issued either by polling the BMC's Status Register (polled mode) or through an interrupt service routine (interrupt-driven).

- Polled Mode. After loading the parametric registers (if required), the software driver issues the desired command, checks to see if the command was accepted, and then continuously polls the BMC's Status Register for an Op Complete indication.
- Interrupt-Driven Mode. After loading the parametric registers (if required), the software issues the desired command, checks to see if the command was accepted, and then waits for the interrupt to occur at successful command completion. Note that the interrupt Enable (Normal) bit must be set in the Enable Register to allow the BMC to generate the interrupt at successful command completion. Also, since an interrupt is not generated if the intended operation fails, additional provisions must be included to avoid "hanging" the system while waiting for the command completion interrupt. Error interrupts are reported according to the settings of the three "error correction" bits in the Enable Register.

Another important fact to note is that the BMC does not support the typical two-way handshaking common to most interrupt-driven peripheral devices and that interrupts from the BMC are cleared (or acknowledged) either when a subsequent command is issued or when the Register Address Counter (RAC) is written with the modifier bit (bit D5) set to "one" and with bits D3 through D0 set to "zero"; interrupts are not cleared by reading the BMC's Status Register.

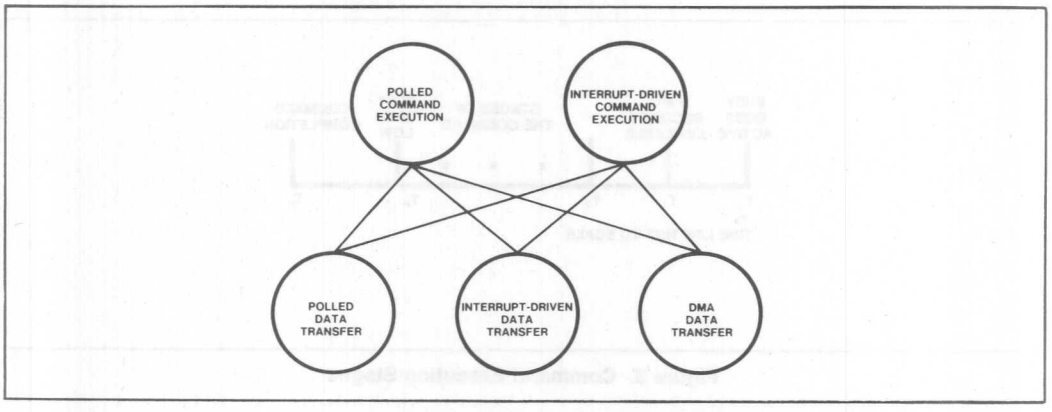


Figure 2. Bubble System Operating Modes

Considerations For Polled Command Execution

The operation of the BMC (and in particular, the BMC's Status Register) during polled command execution is the key to software driver development. Figure 3 shows the activity of the BMC at various stages during a typical command execution sequence. The disussion of the BMC's Status Register is centered around the status bits associated with command execution (i.e., the BUSY, OP COMPLETE, and OP FAIL status bits) although as will be explained, additional Status Register bits can have an effect on the state of the BUSY bit.

Before the command is sent at time T0, the BUSY bit is clear and, in fact, must be clear in order for the BMC to accept a new command (other than an Abort command). Sometime between T0 and T1, the BUSY bit is set to indicate that the command has been accepted and that command execution has begun. Note that all software examples in the remainder of this chapter include a check to ensure that the BUSY bit has been set after a command has been issued.

If the Status Register is examined between T1 and T2, the status byte value will be 80H (BUSY bit set). Depending on the command issued, the FIFO AVAILABLE bit may set at T2 if it is necessary for the host to initiate the transfer of data (see "polled data transfers" later in this section). Whether or not the FIFO AVAILABLE bit is set, the only normal status byte values between T2 and Tn-1 are 80H or 81H (BUSY and FIFO AVAILABLE).

At time Tn (completion of the command), the Status Register indicates the success or failure of the command. Generally, the BUSY bit is cleared at this time (indicating that command execution has terminated); an exception occurs if the FIFO still contains 22 or more bytes of data at command completion (i.e., when the BMC completes its internal microcode routine). During execution of read commands, it is possible for the host to leave data within the FIFO, and if 22 or more bytes are remaining, the BUSY bit will remain set even though command execution has been completed.

If the command is executed successfully and if the FIFO is empty, the status byte value will be either 40H (OP COMPLETE) or 42H (OP COMPLETE and PARITY ERROR). If data was loaded into the FIFO during the interval between T2 and Tn-2, the possible status byte values will be:

41H or 43H if the FIFO contains less than 22 bytes
C1H or C3H if the FIFO contains 22 or more bytes

In the case of unsuccessful command execution, several additional status byte values again are possible depending on the command being executed. Generally, the OP FAIL bit is set in the Status Register along with additional bits that indicate the nature of the failure (e.g., TIMING ERROR or PARITY ERROR). As with successful command execution, the BUSY bit is cleared when command execution is completed unless the FIFO still contains 22 or more bytes.

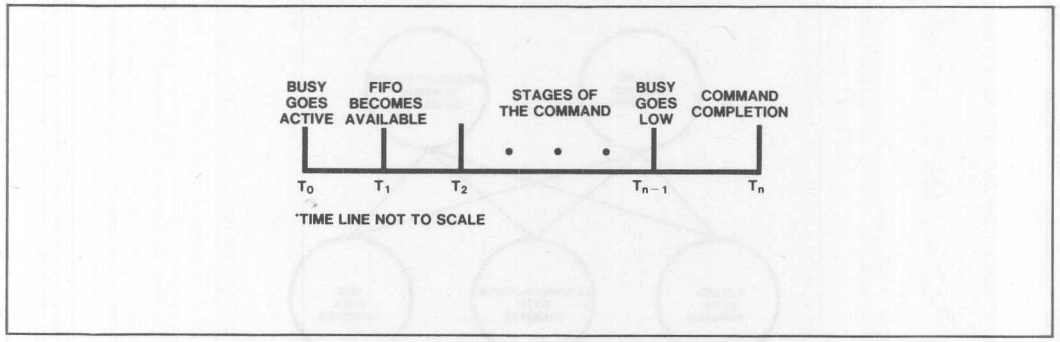


Figure 3. Command Execution Stages

Considerations For Interrupt-Driven Command Execution

As previously mentioned, when the Interrupt Enable (Normal) bit is set, an interrupt occurs only when a command is executed successfully as indicated by the setting of the Op Complete bit in the Status Register. If the Interrupt Enable (Error) bit is not set in the Enable Register and an error occurs that causes the operation to fail, an interrupt will not be generated. The user system must take this fact into account and provide either a fail-safe timer in hardware or a timeout counter in system software to guard against the possibility of "hanging" the system while waiting for an interrupt.

The following possible conditions are applicable to interrupt-driven operation:

1. After issuing a command, the host processor halts and waits for the interrupt to occur. Under this condition, the system will hang if the expected interrupt never occurs. A fail-safe timer connected to a low-level input of an interrupt controller (e.g., an Intel 8259A) would eliminate this problem.
2. In a "true" interrupt driven system where the host processor performs other tasks while the bubble command is being executed, the fail-safe timer in hardware again would be used although it is possible to use a software timeout counter. When operating in the non-DMA data transfer mode (polled or interrupt-driven data transfers), an interrupt service routine would be used to complete the data transfer and then either would wait for the command interrupt or would continuously poll the Status Register until command execution was successfully completed or until a software counter timed out (indicating unsuccessful command execution). However, devoting an excessive amount of time to polling by the host defeats the intention of an interrupt-driven system and reaffirms the need to provide a "watchdog" timer in hardware. In the DMA data transfer mode, the logical approach to detecting when execution of a command has failed is to provide a fail-safe timer in hardware that generates a timeout interrupt to the host.

An additional consideration common to all three data transfer modes is the fact that when an interrupt occurs in response to the successful execution of a command, data still may be present in the BMC's FIFO. If the interrupt service routine immediately acknowledges receipt of the command completion interrupt at the source (i.e., at the BMC), any data remaining in the BMC's FIFO when the interrupt is cleared would be destroyed. To eliminate this potential problem, an intermediate level of interrupt control would be required (again an Intel 8259A) to allow the interrupt routine to complete the transfer. The details described in the "Interrupt-Driven Data Transfer Mode" section will help to clarify the interrupt service routine requirements necessary to avoid leaving any data in the FIFO at command completion.

Data Transfer Modes

As mentioned at the beginning of this section, both polled and interrupt driven command execution can support any of the three data transfer modes. Regardless of the data transfer mode used, the basic operation of the BMC is the same for each data transfer mode. During all data transfers (i.e., during execution of a Read Bubble Data or Write Bubble Data command), the BMC continues to transfer pages of data until the page count in the Block Length Register is satisfied. Since the BMC's FIFO cannot hold a complete page of data, the host processor is required to "keep up" with the BMC as it continues to read data from, or to write data to, the MBM until all data has been transferred. At the beginning of read/write command execution, a "seek" operation is performed to locate the designated page. Once the addressed page is located within the MBM, the read data transfer begins (a write data transfer does not begin until at least two bytes of data have been loaded into the FIFO).

Polled Data Transfers

The polled data transfer mode is the most time-consuming in terms of processor overhead while at the same time is the easiest mode to implement. To support the polled data transfer mode, it is essential that the Status Register bit definitions be clearly understood and, in particular, the function of the FIFO AVAILABLE bit, since the interpretation of the bit changes according to the direction of the transfer.

From an operational viewpoint, the FIFO AVAILABLE bit acts as a gate for the FIFO-handling software. During a bubble write operation, if the FIFO AVAILABLE bit is set, there is room for additional data, and if the FIFO AVAILABLE bit is clear, the FIFO is full. During a bubble read operation, if the FIFO AVAILABLE bit is set, data has been placed in the FIFO by the BMC, and if the bit is clear, the FIFO is empty.

The BUSY bit indicates when the controller is in the process of executing a command. As previously described, the BUSY bit is set within a few microseconds following receipt of the command and remains set until the operation is completed (successfully or unsuccessfully). Remember that since the BUSY bit is internally gated with the BMC's DRQ output signal, it is possible during a read operation to obtain such logically exclusive status indications as BUSY and OP COMPLETE if the host fails to empty the FIFO following comand execution (during non-DMA data transfers, the DRQ signal acts as a "half full/half empty" flag for the BMC's FIFO).

Later in this section, a basic polled data transfer mode read/write routine is flowcharted. Note that to allow the status byte to be considered valid only after the BUSY bit is cleared and to avoid concurrent setting of the BUSY and OP COMPLETE bits at command completion, a running byte counter is implemented in software to ensure that all bytes have been removed from the FIFO. While the FIFO AVAILABLE bit alone can be used to gate the data to or from the FIFO, a byte counter is required to determine when the specified number of bytes has been transferred. Also note that a FIFO Reset command is sent prior to issuing the Write Bubble Data command as a "safety measure." Although issuing the FIFO Reset command is not mandatory, resetting the FIFO is recommended if there is any doubt regarding the state (emptiness) of the FIFO prior to initiating command execution.

Interrupt-Driven Data Transfers

The interrupt-driven data transfer mode requires less processor overhead than the polled data transfer mode (the polling wait time is eliminated) and also allows the data to be transferred in blocks rather than in individual bytes. The actual data transfer rate is fixed; the benefits of this mode only can be realized if the interrupt service routine is efficient (fast) enough to allow additional time for processing other tasks and is based on the host processor's execution speed. Accordingly, if the interrupt-driven data transfer mode is selected, make sure that the additional hardware and software required provide the desired performance increase.

The interrupt-driven data transfer mode is based on the fact that the BMC's DRQ line doubles as a "FIFO half full/FIFO half empty" indicator when the BMC is not in the DMA Mode. Physically, the DRQ line is connected either directly to the host's interrupt input or to an interrupt controller as the interrupt source for the data transfer. When an interrupt occurs, the host processor transfers a block of data to or from the FIFO in a single burst.

The recommended size of the data block transferred is 22 bytes. This block size was selected because transferring 22 bytes, and not 20 bytes (half of the FIFO), accounts for the two additional bytes held in the BMC's internal FIFO input and output latches while also optimizing the buffering capability of the BMC's FIFO. Since 22-byte block transfers rarely are exact multiples of the total number of bytes transferred, an efficient method must be devised to transfer the last remaining bytes of the transfer. While several methods are possible, the following method ensures that an interrupt is issued to transfer all of the "remainder" bytes (i.e., less than 22 bytes never will be "left" in the FIFO to prevent the DRQ line from going active and initiating the last block transfer).

As an example, assume that a two-page (128 byte) read or write data transfer is to be performed. The initial DRQ interrupt indicates that at least 22 bytes should be transferred. However, the first block transferred would be 18 bytes that correspond to the "remainder" bytes (128 mod 22 is 18). By first transferring 18 bytes, five subsequent interrupts occur (each initiating a 22-byte block transfer) to guarantee that no additional polling of the Status Register is required to transfer any "leftover" bytes. The calculation of the remainder count used for the first execution of the interrupt service routine is the responsibility of user software.

DMA Data Transfer Mode

In terms of a bubble system, direct memory access or DMA is the transfer of data between system memory and the BMC without host processor assistance or intervention. Since host processor involvement with the actual data transfer is not required, the overhead associated with software controlled (non-DMA) data transfers is eliminated.

Overall system performance is the primary factor for considering the DMA data transfer mode. That is, if the overhead associated with the available software-controlled data transfer modes degrades system performance to an unacceptable level, the implementation of DMA may prove to be the ideal solution. However, to support the DMA data transfer mode, additional hardware is required (e.g., a programmable DMA controller).

From a system perspective, the DMA hardware definition will dictate the software requirements. In a fixed system, discrete devices may be configured to perform a majority of the data transfer operations, including memory addressing, transfer direction, and terminal count (i.e., the number of bytes to be transferred).

The implementation of a programmable DMA controller such as an Intel 8257 or 8237 provides additional flexibility by allowing DMA to be performed under program control. Since it is not possible to describe all possible configurations, the BMC's operation during DMA transfers is described followed by a specific software example using an Intel 8257 DMA Controller.

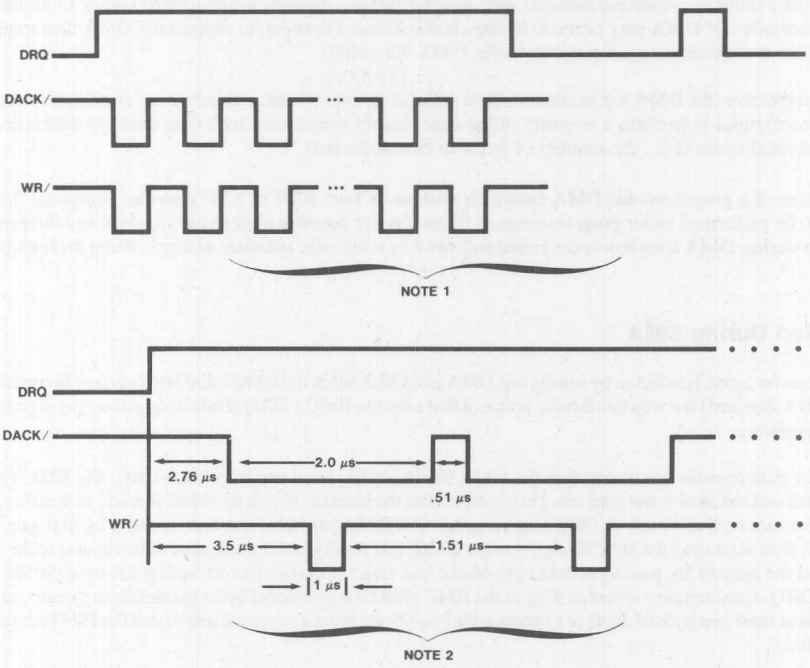
BMC Operation During DMA

The DMA data transfer mode is selected by setting the DMA ENABLE bit in the BMC's Enable Register. In the DMA mode, the BMC supports a standard two-way handshake protocol that uses the BMC's DRQ (Data Request) output signal to interact with the DMA hardware.

To initiate a DMA data transfer (assuming that the DMA hardware has been properly initialized), the BMC's parametric registers are loaded and a read or write command is issued to start the transfer. When the BMC is ready to transfer a data byte to or from system memory, it activates its DRQ output signal. The DMA hardware responds to DRQ by first gaining control of the system bus, then activating the DACK/ signal to the BMC, and finally making a read or write request to the BMC. The BMC responds to the request by placing a data byte on the bus (read operation) or accepting the byte on the bus (write operation). The DRQ signal remains active as long as the BMC either has additional bytes to transfer to system memory (and the FIFO contains at least one byte of data) or expects additional bytes from system memory (and the FIFO contains at least one empty location).

Referring to figure 4, when a Write Bubble Data command is issued, the BMC continuously requests (holds DRQ active) bytes from system memory until the FIFO is filled (i.e., data initially is transferred in a "burst" mode). Once the FIFO is filled, the BMC issues subsequent DRQ requests on a byte-by-byte basis until the last bytes have been transferred. When command execution is complete, the BUSY bit is cleared and the OP COMPLETE bit is set in the Status Register. If the INTERRUPT ENABLE (NORMAL) bit is set in the Enable Register, an interrupt is generated.

When a Read Bubble Data command is issued, the BMC activates DRQ only after the first data byte has been assembled and placed in the FIFO. While the DMA hardware is responding to the DRQ, additional bytes enter the FIFO at a minimum rate of 80 microseconds per byte (the single-MBM system transfer rate; the transfer rate increases according to the number of MBMs operated in parallel). Typical DMA response times usually are fast enough to take each byte before the next byte arrives, and data is transferred on a byte-to-byte basis.



NOTES:
1. THE DATA TRANSFER SEQUENCE INITIALLY OPERATES IN A "BURST" MODE (TYPICALLY 44 TO 46 BYTES ARE TRANSFERRED).
2. THE TIMING SHOWN IS ONE EXAMPLE OF VALUES OBTAINED DURING NORMAL OPERATION. THE VALUES MAY VARY IN YOUR SYSTEM PROVIDED THEY ARE WITHIN SPECIFICATIONS.

Figure 4. DMA Handshake Timing During Read/Write Command Execution

START-UP PROCEDURES

Whenever power is applied to the bubble memory system, a powerfail reset circuit is activated to satisfy specific power/timing sequences required by the BMC. All bubble system designs must include a powerfail reset circuit to ensure proper initialization of the bubble system.

The primary function of the powerfail reset circuit is to ensure that the bubble memory system powers up correctly. A built-in hardware time delay allows the BMC to complete its internal power-up sequence prior to enabling the additional support components; a software time delay is needed before any commands can be issued to the BMC. Following the delay, the first user communication with the BMC must be an Abort command whether the power-up is a cold start (application of power) or a warm restart routine (a RESET/ pulse applied to the 7220-1 BMC). Note that the status register should not be considered valid until the Abort command has been issued. A complete power-up flowchart, including initialization, is shown in figure 5.

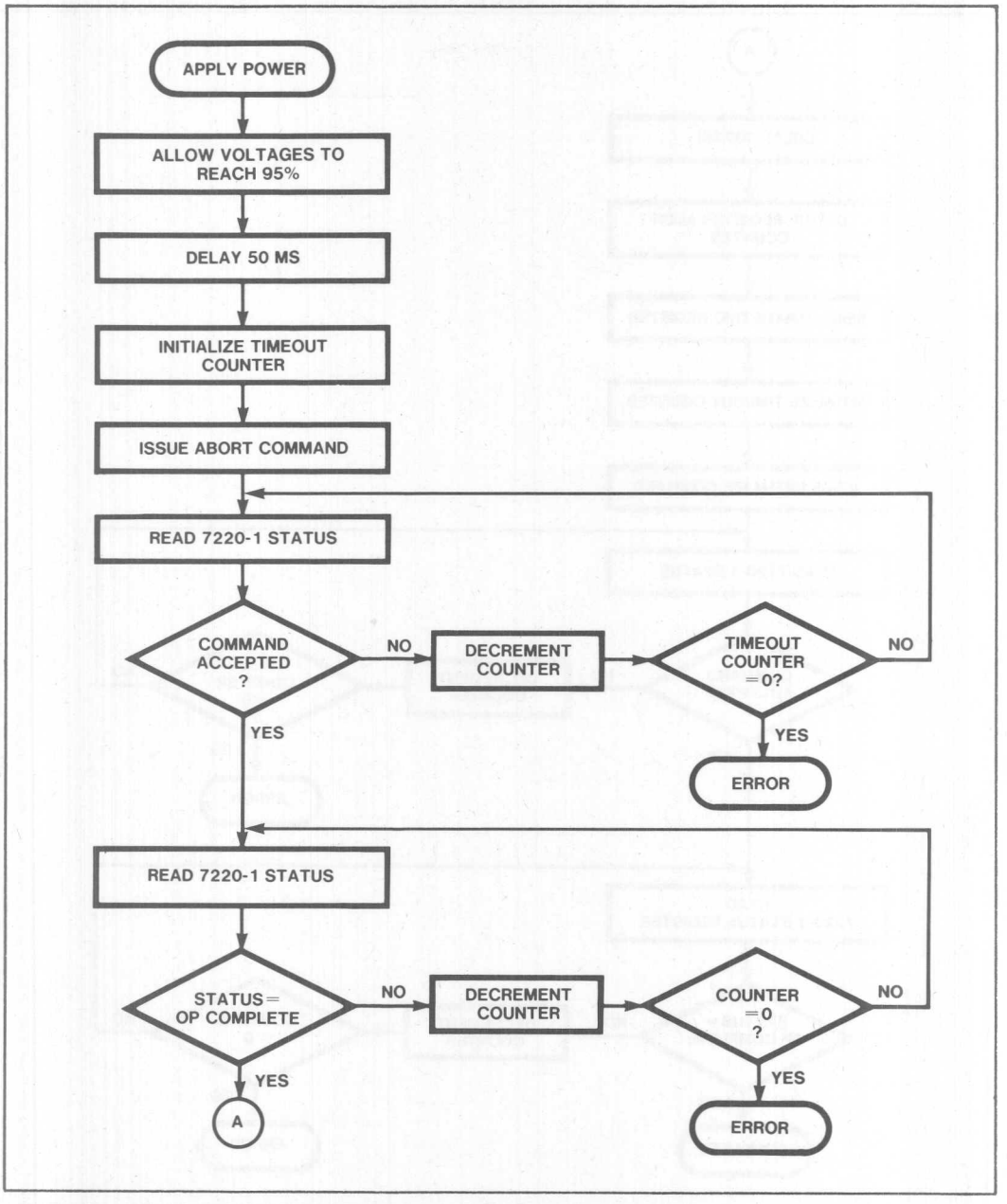


Figure 5. Power-Up Sequence (Polled Command Execution)

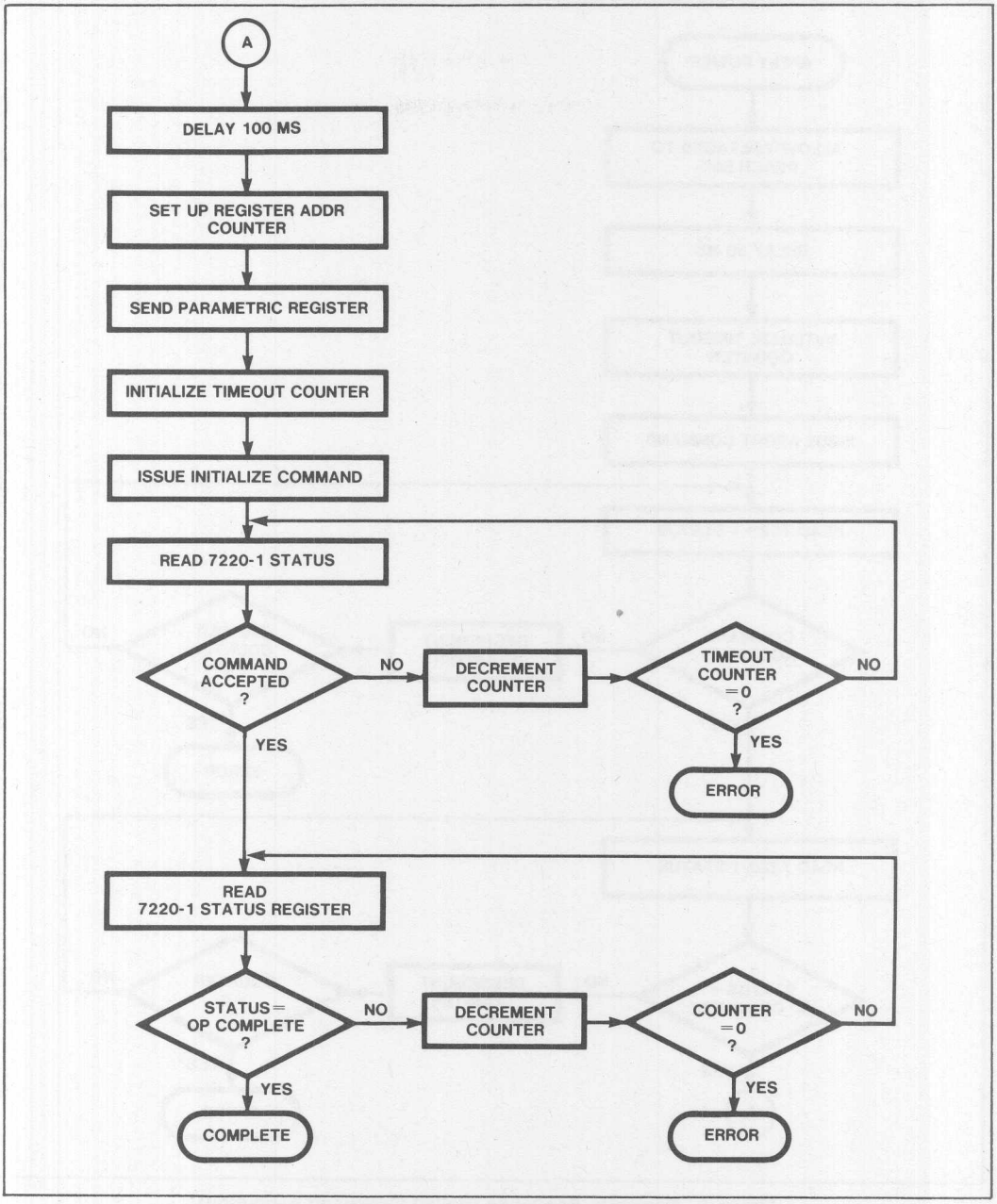


Figure 5. Power-Up Sequence (Polled Command Execution) (Cont.)

Initialization

Following successful execution of an Abort command, the user must initialize the bubble memory system using the Initialize command. The Initialize command requires that the parametric registers first are loaded with specific values. The software flowchart (figure 5) shows one example for polled command execution without error correction for a one-bubble system (i.e., the BPK 72 or iSBX 251). The Block Length Register always must be loaded with the values shown while the MBM Group Select bits in the Address Register always must select the last MBM in the system. For your particular application, set the appropriate bits in the Enable Register prior to issuing the Initialize command. Note that error execution mode changes require re-initialization of the bubble system.

The flowchart example polls the Status Register to see whether or not the initialization was successful. The OP COMPLETE bit indicates success. If an initialization problem occurs, the TIMING ERROR and OP FAIL bits will be set in the Status Register.

In an interrupt-driven system, the interpretation of the INT (interrupt) line depends on the setting of specific bits in the Enable Register. If INTERRUPT ENABLE (NORMAL) is set in the register, on the *successful* completion of command execution, the INT line will activate on the user interface. If INTERRUPT ENABLE (ERROR) is set in the register, the INT line will activate when an error condition occurs (in this case a TIMING ERROR). The user should devise interrupt service routines that can differentiate between the two interrupt sources by polling the Status Register. It is apparent from this discussion that a system that relies solely on interrupts may find it necessary to incorporate a watchdog timer (timeout counter interrupt) depending on the selection of the interrupt enable bits. A timeout counter avoids the possibility of never receiving an interrupt if a command is *unsuccessful* (this condition is possible if only the INTERRUPT ENABLE (NORMAL) bit is set).

If a system uses the BMC's DRQ signal for data transfers (DMA or interrupt mode), special precautions are necessary during Intialization. First, it is recommended that the DMA Enable bit in the Enable Register be disabled (i.e., set to 0) for the Initialize command. Once the Initialize command is successfully completed, the DMA Enable bit is enabled for the subsequent DMA data transfer.

When the BMC is in the non-DMA mode, the DRQ pin acts as a half full/half empty flag for the BMC's FIFO. Since the FIFO is used to temporarily store bootloop data during execution of the Initialize command, the DRQ pin is toggled. User-written software should therefore take appropriate precautions with respect to the DRQ pin during initialization (i.e., mask all interrupts, disable DMA controllers, etc.).

The Initialize command establishes a "working subset" of the bootloop information from the bubble and places this information into the bootloop registers of the FSA. A masking algorithm within the BMC reduces the total of 320 possible good storage loops to a working subset of 270 or 272 good loops (error correction dependent).

If the bootloop information in the MBM is incorrect, the system can inadvertently appear to be initialized correctly. As a one-time check to ensure that the system has been initialized correctly, simply read the bootloop registers from the FSA and count the number of "1's."

The count should yield:

- 270 "1's" out of 320 with error correction implemented.
- 272 "1's" out of 320 without error correction.

NORMAL OPERATION EXAMPLES

The following software flowcharts outline the important considerations and provide examples of each mode of operation previously discussed. First, figures 6 and 7 show the two possible command execution techniques. Note that in each case, the appropriate additional software must be inserted according to the data transfer mode selected. Figures 8 through 10 detail each data transfer mode. For any commands that do not involve the transfer of data (i.e., Abort, FIFO Reset, etc.), no additional software is required to be inserted in the command execution examples.

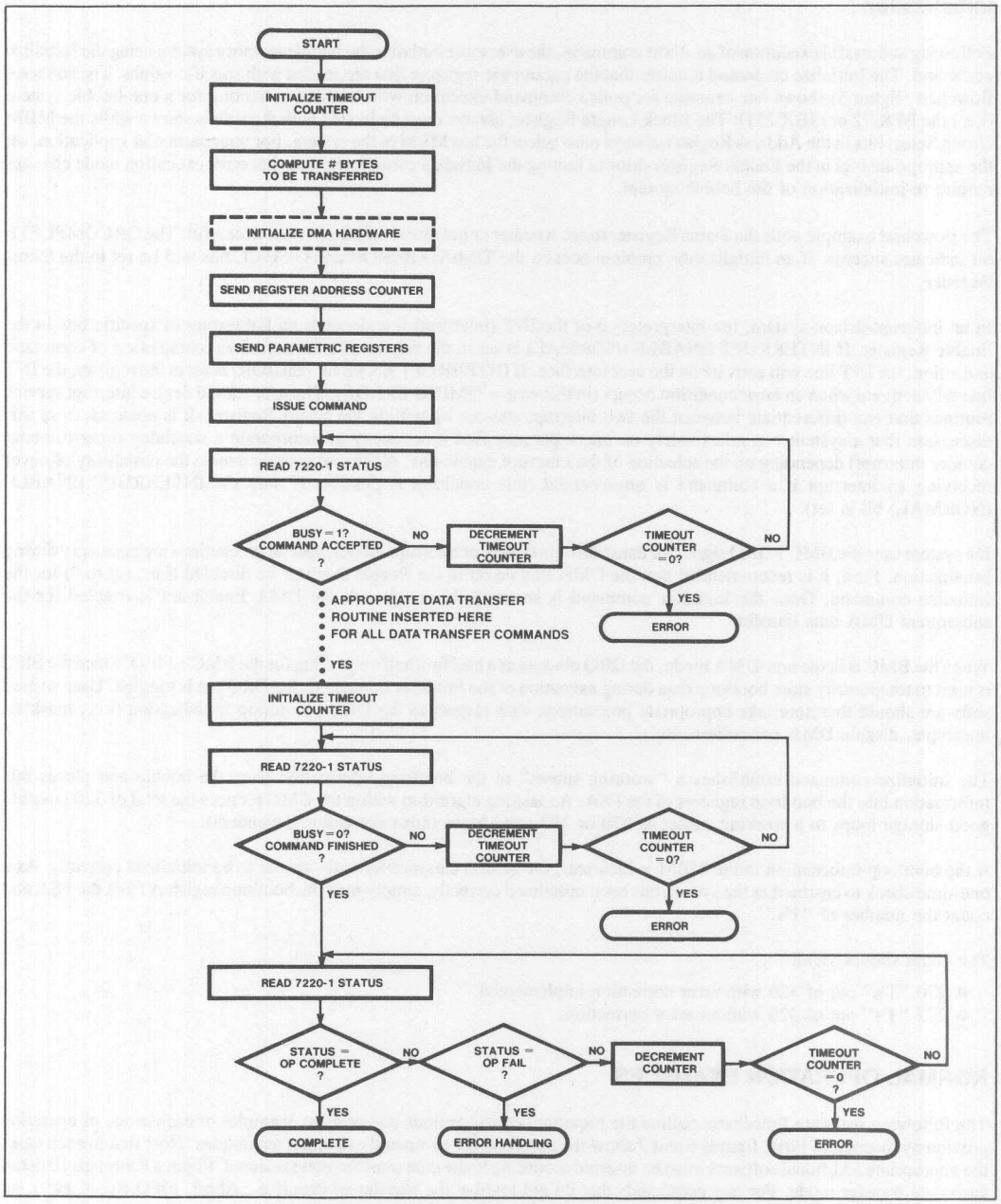


Figure 6. Polled Command Execution Routine

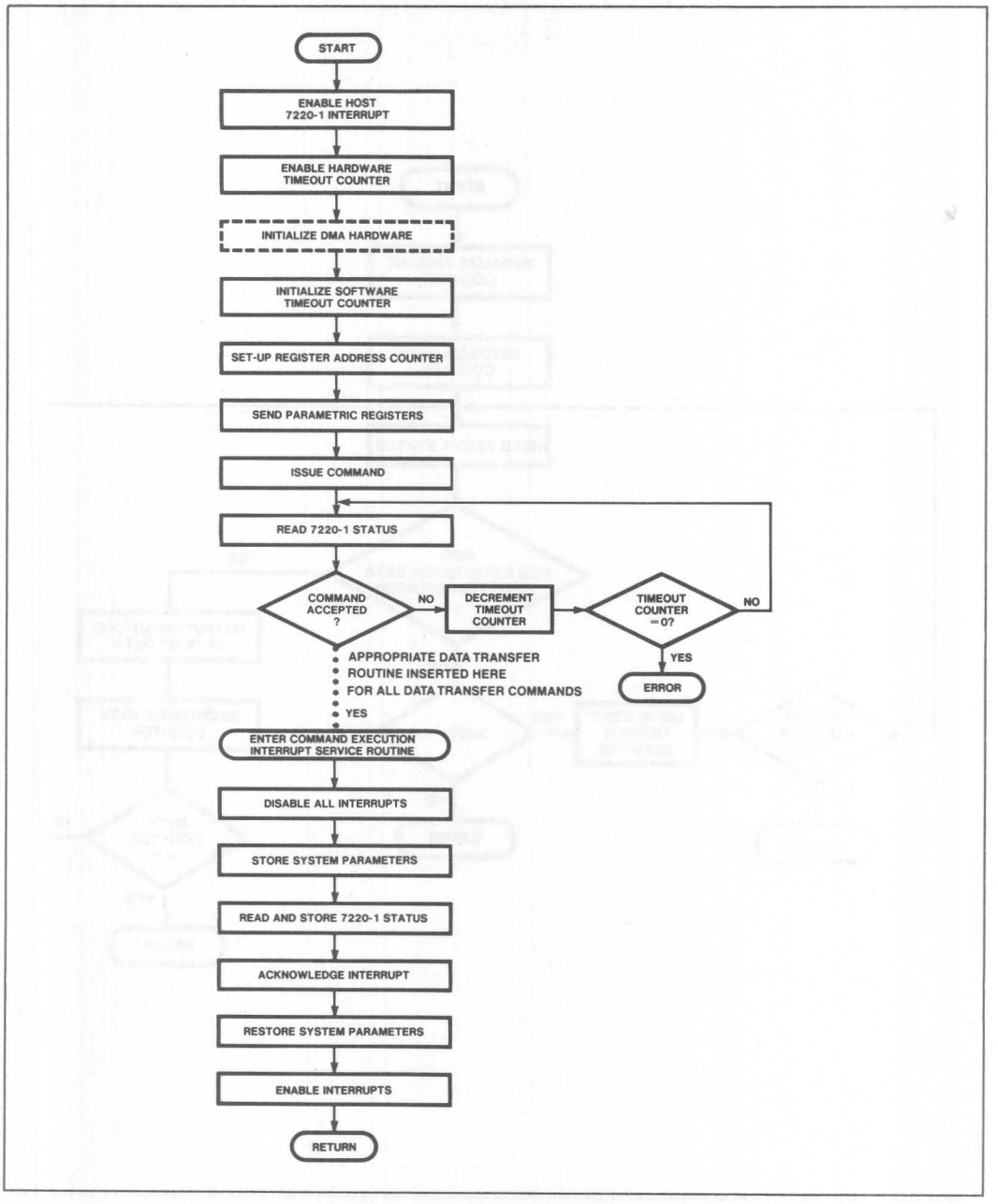


Figure 7. Interrupt-Driven Command Execution Routine

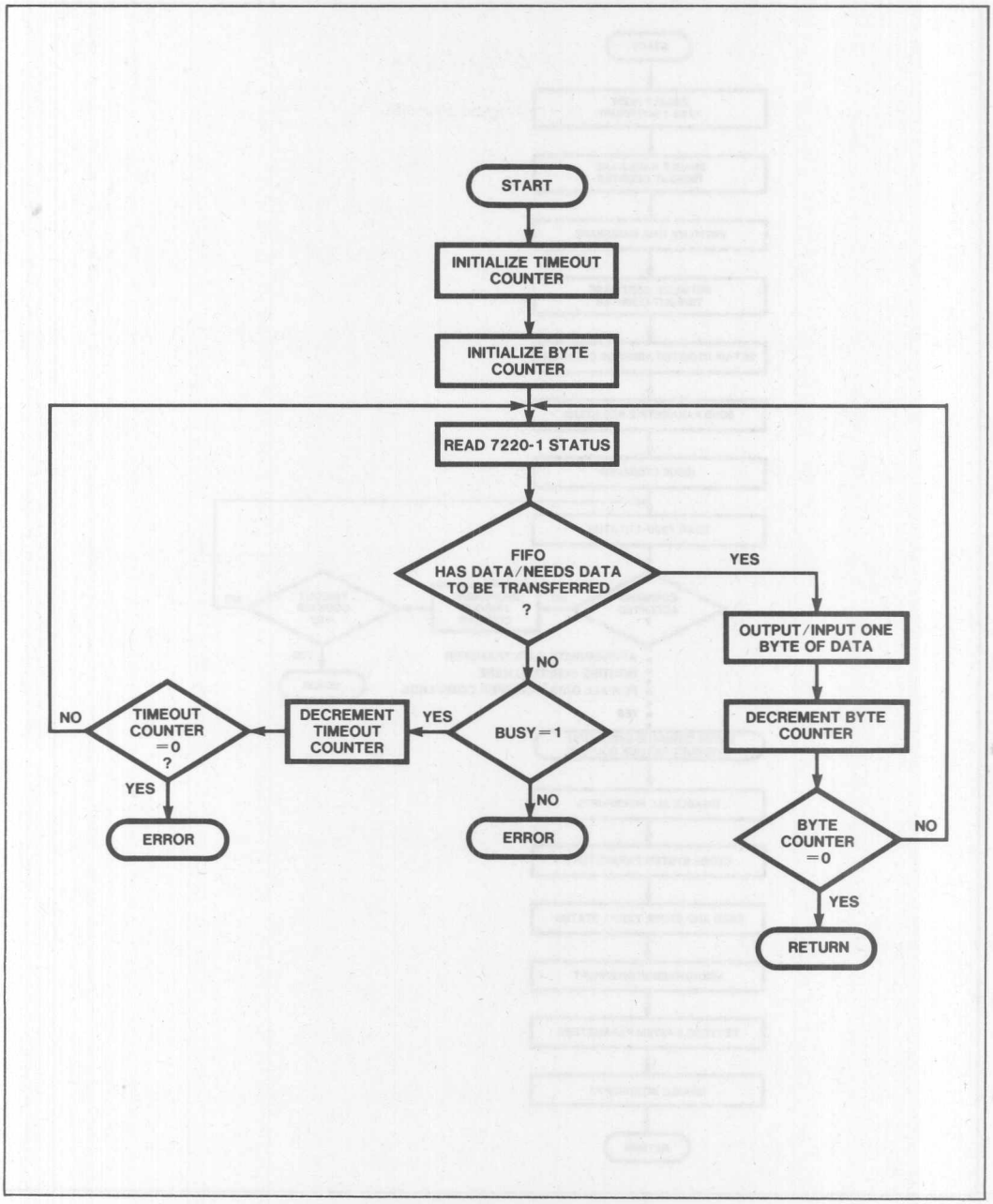


Figure 8. Polled Data Transfer Routine

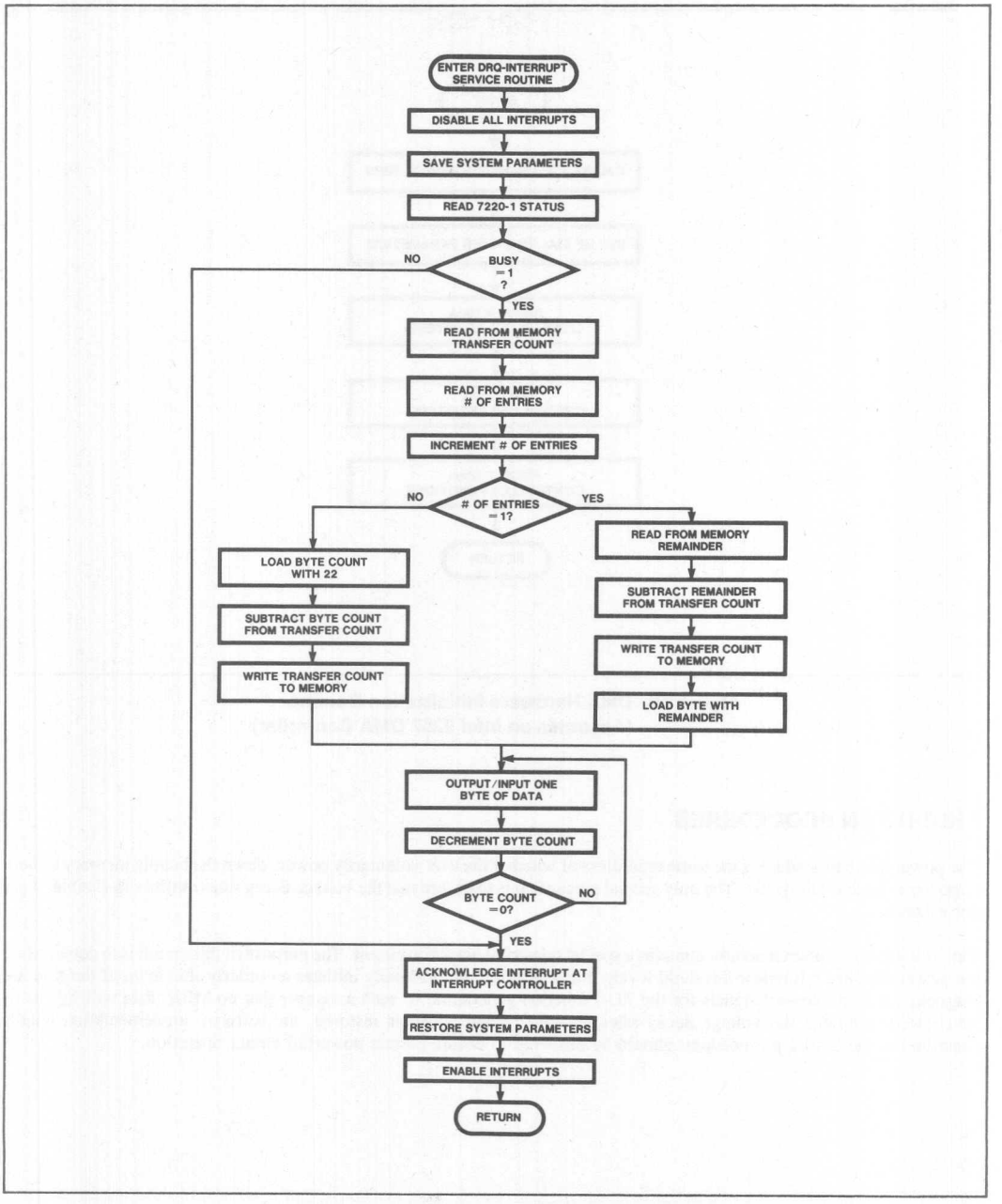


Figure 9. Interrupt-Driven Data Transfer Routine

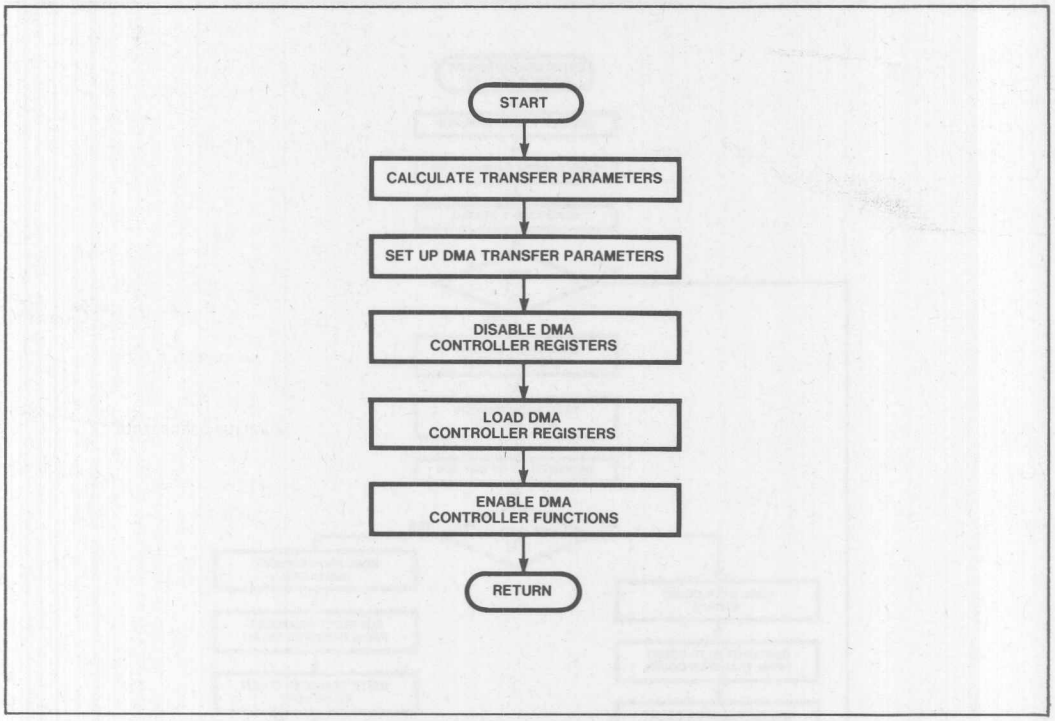


Figure 10. DMA Hardware Initialization Routine (Assumes an Intel 8257 DMA Controller)

SHUT-DOWN PROCEDURES

The power down procedure is the same regardless of whether the user voluntarily powers down the bubble memory system or power is inadvertently lost. The only special precaution is to ensure that the voltage decay rates outlines in Chapter 4 are not exceeded.

The 7230 Current Pulse Generator contains a special powerfail detection circuit. The purpose of this circuit is to detect when the power supplies fall below threshold levels. Such an event automatically initiates an orderly shutdown of the rotating magnetic field and control signals for the 7110 function generators, in such a manner that no MBM data will be lost or invalidated, provided the voltage decay rates are met. When power is restored, the software implementation details described in the Start-Up procedures should be observed to ensure correct powerfail circuit operation.